Discovering Epistatic Interaction Features via High Performance Machine Learning

Qiao Wang

Submitted in total fulfilment of the requirements of the degree of

Doctor of Philosophy

School of Computing and Information Systems
THE UNIVERSITY OF MELBOURNE
ORCID 0000-0002-6085-8135

May 2021

Abstract

GENOME-wide Association Studies (GWAS) are increasingly popular in screening the genetic architecture of human genome. For a typical GWAS study, half a million genetic variants can be genotyped in the form of single nucleotide polymorphisms (SNPs) for thousands of individuals, enabling the association analyses at the population level. The genetic profiling using GWAS raises the possibilities of presymptomatic diagnosis, preventive interventions and finally, precision medicine.

Consequently, the identification of informative SNPs from GWAS studies becomes a main concern in bioinformatics research. However, such identification poses significant statistical and computational challenges due to the high dimensionality of GWAS data. The curse of dimensionality restricted the power of statistical tests and limited the practicability of data processing.

Currently, the major focus of GWAS analyses is to identify SNPs that are highly associated with the traits using tailored statistical tests. For example, commonly used linear models can be applied to select independent SNPs with strong marginal effects. A more complicated scenario is epistatic SNP interactions in which each SNP has little marginal effect, but jointly have non-negligible synergistic effects. In such a case, association tests are designed to detect interactions (e.g. SNP-pairs, triplets, etc) rather than individual SNPs. However, the statistical tests targeting SNP interactions are insufficient by themselves. The identified SNP hotspots can be highly associated with the traits, but their roles in disease prediction remain unknown since they are not necessarily effective predictors in classifying disease-group (Cases) and healthy-group (Controls) in GWAS. Therefore, it is also necessary to identify the predictive ones before further analyses.

While it is relatively easy, in theory, to detect SNP features with marginal effects us-

ing linear models, the identification of discriminative epistatic interactions is particularly hard. For an interacting SNP-pair, genotypes of a single SNP may have little predictive power by themselves unless combined with the genotypes of another SNP to form a nonlinear interaction pattern for effective Case-Control discrimination. The learning of interaction features for classification, however, is a great challenge due to the following two bottlenecks: 1) From computational point of view, exhaustively analysing all bivariate interactions for a GWAS data containing n SNPs requires total $n \times (n-1)/2$ statistical tests, usually resulting in an excessive long time of computation which is infeasible for data processing; 2) The lack of a simple predictive model that stably and effectively learns various nonlinear interactions for Case-Control classification. Consequently, the efficient and effective learning of interaction features for Case-Control classification in GWAS is questioned by research community. In this thesis, we address the challenges by proposing a two-stage analytical framework as the technical contribution.

Firstly, a GPU-based filtering platform is proposed to detect robust interactions via re-samplings in association studies. Customised statistical tests can be implemented via the GPU interface and SNP-SNP interactions are analysed in a fast exhaustive manner at each re-sampling trial. Robust interactions can then be determined by the re-sampling consistency. Original test methods in literature received significant speed-up via the proposed platform. In addition, more efficient GPU algorithms are proposed to further improve the efficiency of re-sampling trials for robust interactions detection.

Secondly, a shallow neural network model is proposed, enabling the Case-Control classification and predictive interactions selection at the same time. The simple method identifies meaningful features in a reasonable way, thus being interpretable, to some extend, for human understanding of learned features. It is shown to be able to realise various combinations of nonlinear discriminative patterns.

Lastly, the two-stage filtering framework combines two proposed methods to effectively learn the robust interaction features for Case-Control classification. In this thesis, the breast cancer GWAS studies are used for demonstration. The topic mainly focuses on computing and machine learning aspects, aiming at introducing high performance data processing tools to genetic epidemiologists and other domain experts.

Declaration

This is to certify that

1. the thesis comprises only my original work towards the PhD,
2. due acknowledgement has been made in the text to all other material used,
3. the thesis is less than 100,000 words in length, exclusive of tables, maps, bibliographies and appendices.

Qiao Wang, May 2021

Acknowledgements

At the end of this Ph.D. study, I would like to thank all people who have helped during this long journey of research.

First of all, I would like to thank my supervisors Dr. Aaron Harwood, Dr. Miroslaw Kapuscinski and Prof. John Hopper who have been guiding me through the research and provided countless help in the past few years. They have been patient and generously offered their insights. Throughout the research project, and especially when difficulties arose, they have been trying to encourage me, lower my burden and help me to focus on the research. I have learned a lot from them not only technical knowledge but also the way of thinking, planning and conducting research. Without them, I would not even have this research training opportunity, and this thesis would not have been possible.

In addition, I would like to thank Prof. Christopher Leckie, the chair of advisory committee. Although Prof. Leckie is not a supervisor, he offered valuable advice in machine learning. Moreover, I gradually learn from him how to systematically carry out the research and how to make decisions in a proper way.

Also, special thanks to Dr. Cheng Soon Ong. Cheng Soon played an importing role in my early research career. He is very knowledgeable and his wisdom stimulated my curiosity about machine learning. He has been supporting me since the early years of research till the end of my candidature.

Moreover, I am grateful for my previous supervisor Dr. Adam Kowalczyk. Adam is the one who brought me into the research in genomics. During the four years at NICTA and early years of Ph.D. study, I had gained fundamental knowledge and experiences under his guidance. Because of him, I had an opportunity to step into the genomics research, to attend conferences and participate in publishing paper.

Next, I would like to thank collaborators and previous colleagues Richard Campbell, Sylvia Young, Benjamin Goudey, Andrew Isaac, Fan Shi, David Rawlinson and Herman Ferra for their generous help, contributions and valuable discussions.

Here, I would also like to acknowledge the following support:

- Funding from Prof. John Hopper: NHMRC project grant 12/14 - Hopper_Contract #34016, Project Num 057586;
- Google Australia for conference support;
- NICTA for conference support;
- NVIDIA for a Tesla K40c GPU donation;
- LIEF HPC-GPGPU Facility hosted at the University of Melbourne. This Facility was established with the assistance of LIEF Grant LE170100200.

Special thanks to the Australian Government and the University of Melbourne for offering me abundant resources and a place in this Ph.D. course. Also I would like to thank all staff in school of computing and information systems and school of population and global health for their day-to-day help.

Last but not least, I present the greatest respect and thanks to my parents for their endless love and support. I hope they can be proud of this achievement.

Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Overview

Genomics-based diagnostic tests are increasingly popular due to the pervasive sequencing technologies in recent years. Fast genetic tests are applicable at low cost, generating cumulated genomics data for knowledge discovery. By identifying and understanding key genetic elements from data, potential drug targets may be located for treatment. As the starting point, Genome-wide Association Studies (GWAS) are widely used for genotyping genetic variants for thousands of individuals, enabling the genetic profiling at the population level. Effort has been dedicated to mining valuable genetic variants from GWAS data, trying to discover disease-associated variants. Once they are determined, related functional genes and other elements may be of interest to further bioinformatics analysis and possible biological interpretation. However, the learning of meaningful genetic variants from GWAS data is non-trivial.

An important criteria of identifying informative variants is to test their predictive power in Case-Control classification. Among many factors, one of the major challenges is raised by epistatic gene-gene interactions, for which no individual gene/variant is an effective predictor by itself unless they are combined together to form a nonlinear discriminative pattern. The intrinsic nonlinear nature and the diversity of various interaction patterns pose significant methodology and computational challenges, and how to effectively and efficiently detect robust interaction predictors from GWAS studies is the the focus of this thesis.

In this chapter, we give related concepts in GWAS; highlight the challenges raised by

the topic; define the research question and scope; and summarise the contributions for each chapter. We discuss the topic mainly from computing and machine learning point of view throughout the entire thesis, aiming at providing practical computational tools for learning robust interaction features.

1.2 Concepts in GWAS

Related concepts and background knowledge in biology and epidemiology are introduced in this section for understanding the research topic.

1.2.1 DNA and Chromosome

The basic genetic element of human being is known as deoxyribonucleic acid (DNA), a double helix structure in the form of a sequence of *nucleotides*. Each nucleotide contains one of four possible bases: adenine (A), thymine (T), cytosine (C) or guanine (G). Within the double helix, base 'A' on one strand is always paired with base 'T' on the complementary strand while 'C' is always paired with 'G' via hydrogen bonds. This base pair structure is briefly illustrated in Figure 1.1. Segments of these base pair sequences, such as *genes*, are firstly transcribed into *ribonucleic acid* (RNA) and then translated into proteins which become functional elements in cells. This procedure is called *central dogma* [108], showing that the synthesis of proteins is derived from DNA. A long sequence of DNA base pairs is packed around histones into a compact structure called *chromosome*. A human cell contains pairs of chromosomes, with one set given by the father and the other contributed by the mother. A total of 23 homologous pairs (including 22 autosomes pairs and two sex chromosomes, either X or Y) uniquely define a person's genetic makeup.

1.2.2 Allele, Genotype and Phenotype

The genetic codes are huge in volume and complicated in nature. The total number of DNA base pairs is estimated around 3×10^9 in human genome, and the bases differ in the frequency of $1/1000$ between any two individuals [161]. For example, in Figure 1.1,

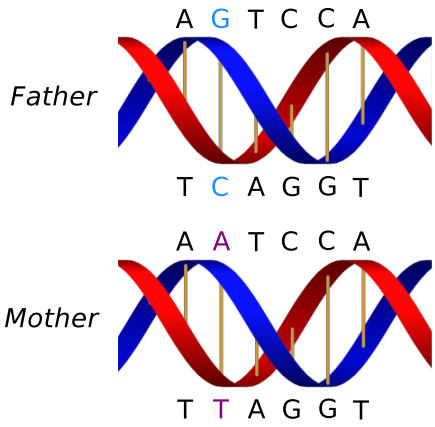

Figure 1.1: A human cell inherits two copies of homologous chromosomes from parents.

the 'G-C' base pair at the second locus of father's DNA differs from the 'A-T' base pair of mother's DNA at the same locus. Therefore, we have two different versions of base pairs here, which are termed *alleles*. Usually, one version is more prevalent in the human population than the other and it is called *major allele* while the less frequent one is usually called *minor allele*. The *minor allele frequency* (MAF) in human population is frequently used to determine the type of genetic variant. In this example, assuming that the major allele is father's 'G-C' and the minor one is mother's 'A-T', and we denote *'A'* and *'a'* to major and minor versions respectively. Consequently, an individual carries one of three possible allele combinations at a locus either 'AA', 'Aa' or 'aa', corresponding to *dominant homozygous*, *heterozygous* and *recessive homozygous* respectively. These allele combinations are called *genotypes*. Therefore, the depicted individual in Figure 1.1 possesses a heterozygous genotype at the second locus of genome.

Phenotype, on the other hand, describes observable traits such as the skin and hair colour. Phenotype is decided jointly by genotype and environment factors [140]. In the context of this thesis, the phenotype refers to the health status of the individuals either Case (disease) or Control (healthy).

1.2.3 SNP and GWAS data

Genetic variations broadly exist in human population. Among others, the most abundant genetic variants are *single nucleotide polymorphisms* (SNPs). A genetic variant at a locus

can be considered as a SNP when the MAF is greater than one percent of the human population at that locus [161].

GWAS assays hundreds of thousands to million-level SNPs for thousands of individuals [85] into Case and Control groups for profiling the genetic architecture. As a result, a typical GWAS dataset is in high-dimension, containing genotype information for each individual at each locus. Usually, dominant homozygous (AA), heterozygous (Aa) and recessive homozygous (aa) are encoded into numerical numbers such as '0', '1' and '2' respectively. An illustration of a GWAS dataset is given in Figure 1.2.

	GWAS	SNP 1	SNP 2	...	SNP N
Cases	Individual 1	1	0	0	2
	Individual 2	0	2	2	1
	...	1	1	2	0
Controls	...	2	0	1	0
	Individual M-1	2	0	1	1
	Individual M	1	2	0	0

Figure 1.2: Each row represents an individual, either Case or Control. Each column represents a SNP and the number denotes dosage of the minor allele. In most cases, GWAS data are high-dimensional and sparse.

1.2.4 Penetrance

Penetrance is the probability $P(D|G)$ of exhibiting a phenotype, such as a disease (D), given a particular genotype or genotype combination (G) [22,100]. A *fully penetrant* model [75] is derived when an individual carrying a specific genotype (e.g. a recessive homozygous 'aa') at a locus will either 100% develop the disease ($P(D|G) = 1$) or not develop at all ($P(D|G) = 0$). For a full penetrant variant, in the context of GWAS data, all individuals holding the deleterious genotypes are categorised into the Case group while none of them can appear in the Control group, indicating that the disease is entirely decided by genotype [137]. However it is less likely the case for complex diseases [22] since high-penetrant variants may be prevented by human evolution process [84], and large number of common low-penetrant variants are hypothesised to be the driven factors behind the

common diseases [84]. As a result, the penetrance function is often a continuous value ranging from zero to one in most scenarios.

1.2.5 Heritability

Since the phenotype of disease is less likely driven by high-penetrant variants but more likely determined by the effects of many low-penetrant variants, then a natural question arises here is to what extent a phenotype variance can be explained by the genotype variance. This question falls into the concept of *heritability*. As a widely-known phenotype example, the human height is estimated 80 $\sim$ 90% heritable, but according to the tiny effects of a set of detected genetic variants, only little more than 5% of heritability can be explained for this trait, summarised in [84].

Heritability (H^2) estimates the proportion of phenotype variance σ_P^2 due to the genotype variance σ_G^2. The formal definition of heritability is divided into two forms, in either broad or narrow sense by [140] as follows:

$$H^2_{broad} = \frac{\sigma_G^2}{\sigma_P^2}, \quad h^2_{narrow} = \frac{\sigma_A^2}{\sigma_P^2}, \tag{1.1}$$

where $\sigma_G^2 = \sigma_A^2 + \sigma_D^2 + \sigma_I^2$. Here, σ_A^2 is the additive effect (measuring the average effect of all alleles at a locus), σ_D^2 is the dominance effect (measuring the interactions between different alleles at a locus) and σ_I^2 is the interaction effect (measuring the interactions between different loci). As can be seen, the narrow sense heritability h^2_{narrow} only considered the additive effect σ_A^2 in the calculation. Due to the technical difficulties, the calculation for non-additive effect is often impractical [146].

On the other hand, solely relying on additive effects from current GWAS studies is insufficient for heritability estimation because the true heritability may be underestimated, which may end up with the *missing heritability* problem. There are possible explanations [85, 156], including, but not limited to: 1) the small effects of existing variants and the complete set of related variants are not known yet; 2) limited capacity in detecting interaction effects between loci. It turns out to be only a small proportion of heritability can be explained for most diseases of interest [48].

1.2.6 Epistatic Interactions

Epistatic interaction or *epistasis* was first introduced by Bateson [7], describing the effect of an allele/variant at one locus of being masked by another locus [23, 99, 141]. As an example, the coat colour of dog (Labrador retriever) was studied in [120, 131] and summarised by [161], showing that the manifestation of the colour phenotype was jointly decided by a pair of variants. Generally from biological point of view, related biomolecules at cellular level may physically interact with each other to trigger the epistasis [101], and an algorithm can help to build the interaction network for a better understanding of the possible mechanisms for relatively simple organisms like yeast [121].

The formal definitions of epistasis can be found in many forms [23]. From biological point of view it refers to the Bateson's version, and we consider interactions when one biological factor's qualitative nature is affected by the status of another one [124]. From the perspective of statistics and quantitative genetics, Fisher [31] defines the epistasis as the deviation from additive effects at multiple loci via statistical models. First, consider the following two scenarios:

- Assuming there are two SNPs SNP_1 and SNP_2 under investigation, and full penetrance is used for simple demonstration. We denote 'A' and 'a' to the wild-type and mutant-type alleles respective at SNP_1, while 'B' and 'b' are used for SNP_2. Imagining, the disease is defined as fully manifested when there exist at least one mutant-type copy 'a' at SNP_1 and regardless how many copies of 'b' at SNP_2. Therefore, $P(D) = 1$ when $SNP_1 \in \{1(Aa), 2(aa)\}$ and $SNP_2 \in \{0(BB), 1(Bb), 2(bb)\}$, and $P(D) = 0$ when $SNP_1 \in \{0(AA)\}$ and $SNP_2 \in \{0(BB), 1(Bb), 2(bb)\}$. This scenario shows that the disease solely relies on the genotypes of SNP_1, and a linear model suffices to determine the phenotype.
- Assuming the disease manifestation is modified, and it is required that at least one mutant-copy at both SNPs to develop the disease. In this case, we have $P(D) = 1$ when $SNP_1 \in \{1(Aa), 2(aa)\}$ and $SNP_2 \in \{1(Bb), 2(bb)\}$, and $P(D) = 0$ when $SNP_1 \in \{0(AA)\}$ and $SNP_2 \in \{0(BB), 1(Bb), 2(bb)\}$, or $SNP_2 \in \{0(BB)\}$ and $SNP_1 \in \{0(AA), 1(Aa), 2(aa)\}$. In this scenario, the disease relies on the presence of mutant alleles at both SNPs, and zero copy of such allele at either locus leads to

the penetrance function equals to zero. This represents a typical epistatic interaction effect [23, 86] which can be detected by a nonlinear model.

In more general sense, statistical methods are used for quantifying the interaction effects. A commonly used one is logistic regression, modelling the binary outcome (the probability of developing a disease $P(D)$) of the predictors [22]:

$$ln(\frac{P(D)}{1-P(D)}) = \alpha + \beta_1 SNP_1 + \beta_2 SNP_2 + \beta_3 SNP_1 SNP_2, \tag{1.2}$$

where β_1 and β_2 represent the additive effects of two individual SNPs, and β_3 represents the interaction term. As can be seen that $\beta_3 \neq 0$ reflects the Fisher's proposal. Consequently, the additive effect-based estimation of narrow sense heritability in equation 1.1 may underestimate the influences of genetic factors, and the understanding to the interaction effects may be critical for the broad sense heritability estimation.

1.2.7 Nonlinearity in Epistatic Interactions

One of the key characteristics of epistatic interactions is the nonlinearity that broadly exists in various interaction models, making those interactions particularly difficult to detect and evaluate.

A number of previously studied interaction patterns are summarised in [75]. Among many models, a typical XOR problem, which is denoted as M78 model, is a standard type of nonlinear interaction. Assuming the minor allele frequency $q = 0.2929$ and the major allele frequency $p = 0.7071$, then the recessive genotype frequency is $q^2 = 0.0858$ and the dominant genotype frequency is $p^2 = 0.5$. According to Hardy-Weinberg equilibrium, the heterozygous frequency is $2qp = 1 - q^2 - p^2 = 0.4142$. Also assume that the full penetrance function is used, then the marginal penetrance of M78 model is illustrated in Table 1.1. The marginal penetrance is calculated as a dot product between the genotype frequencies and the penetrance values. For example, the marginal penetrance values for genotype *aa* and *aA* are both $q^2 \times 0 + 2qp \times 0 + p^2 \times 1 = 0.5$, while the marginal penetrance for *AA* is $q^2 \times 1 + 2qp \times 1 + p^2 \times 0 = 0.5$. The values for the genotypes of the other SNP can be derived in the same way. As can be seen, an equal marginal penetrance

0.5 is shown for all cases, leading to zero main effect.

Table 1.1: This XOR model was previously summarised in [75] as M78 model. The numbers in parenthesis are genotype frequencies, and the numbers in cells are penetrance values. Deleterious and protective genotype combinations are coloured in red and blue respectively.

	bb (q^2)	bB ($2qp$)	BB (p^2)	Marginal penetrance
aa(q^2)	0.0	0.0	1.0	0.5
aA($2qp$)	0.0	0.0	1.0	0.5
AA(p^2)	1.0	1.0	0.0	0.5
Marginal penetrance	0.5	0.5	0.5	

In the absence of the main effect, the independent SNPs are considered not informative by themselves. Univariate analysis will discard them as noise features and they are not effective predictors for a classification algorithm. To provide a nonlinear dataset to a machine learning model, we can derive, from a data point of view, the following Table 1.2 based on Table 1.1, under the assumption of total 10000 samples are allocated to the compartments of the table. We show how such table can be derived by giving two examples of genotype combinations *aabB* and *aaBB* as follows.

Table 1.2: Sample allocation for each genotype combination in penetrance Table 1.1. Assuming an equal sample size 5000 is used for both Cases and Controls. The sample size ratio between Cases and Controls is given in each cell, where red and blue colours represent deleterious and protective cells.

	bb (q^2)	bB ($2qp$)	BB (p^2)	Total
aa(q^2)	0 : 74	0 : 355	429 : 0	429 : 429
aA($2qp$)	0 : 355	0 : 1716	2071 : 0	2071 : 2071
AA(p^2)	429 : 0	2071 : 0	0 : 2500	2500 : 2500
Total	429 : 429	2071 : 2071	2500 : 2500	5000 : 5000

For *aabB*, the sample counts $N_{Cases}(aabB) = P(D|aabB) \times P(aa) \times P(bB) \times N_{Samples}$ $= 0 \times q^2 \times 2qp \times 10000 = 0$, and the counts $N_{Controls}(aabB) = (1 - P(D|aabB)) \times P(aa) \times P(bB) \times N_{Samples} = (1 - 0) \times q^2 \times 2qp \times 10000 = 0.0858 \times 0.4142 \times 10000 = 355$.

For *aaBB*, the sample counts $N_{Cases}(aaBB) = P(D|aaBB) \times P(aa) \times P(BB) \times N_{Samples}$

$= 1 \times q^2 \times p^2 \times 10000 = 0.0858 \times 0.5 \times 10000 = 429$, and the counts $N_{Controls}(aaBB) = (1 - P(D|aaBB)) \times P(aa) \times P(BB) \times N_{Samples} = (1 - 1) \times q^2 \times p^2 \times 10000 = 0$.

The remaining cells are filled in the same way. As can been seen, if we examine each genotype independently, then the number of Cases equals to the number of Controls, disabling the classification ability for a linear model in Case-Control classification. However, if two SNPs are considered jointly, then they become a valid predictor for a nonlinear classification algorithm (e.g. a nonlinear kernel method classifies a test instance into the Case group when encountering a red genotype combination, and classifies into the Control group otherwise). Thus, univariate-based search strategy and linear models are insufficient, requiring tailored statistical tests for interactions and flexible predictive models to realise nonlinear interactions. Two-locus models appeared in various shapes[44, 75], and they are subject to several factors such as the penetrance functions, MAF, heritability, etc. Higher order interactions and the sparsity of data further complicate the situation.

1.3 Significance of the Research Topic

In recent years, genotype technology raises the possibilities of precision medicine and preventive interventions via personal genetic testing. The first step of genetic-based preventions and diagnosis requires an understanding of the underlying genetic variations [85]. To reveal the genetic architecture for diseases, millions of genetic variants are genotyped for thousands of individuals using GWAS. The genetic profiling can then apply to test multiple loci for disease prediction [57]. For Mendelian diseases, a single inherited variant is sufficient to trigger the disease. A typical example of such monogenic disorder is Huntington disease in which the variation in one gene is the major genetic cause. In comparison, the mechanism for most complex non-Mendelian diseases is quite different. They are generally determined by ubiquitous gene-gene and gene-environment interactions, and the interactions of multiple loci play a crucial role in the disease aetiology [13, 32, 44, 49, 57, 92, 141, 146].

One common way of observing the effects of multiple genetic loci is to measure their capacities in disease prediction. For example, one may interest in learning SNP predictors

in Case-Control classification. Effort has been dedicated to improve the classification performance. Even a small improvement may benefit the clinical applications for preventive purpose [48]. However, the gain in prediction is only incremental using low penetrance variants in GWAS data, and the combinations and epistatic interactions of genetic markers are non-negligible factors that need to be addressed for disease prediction [57, 100]. The nonlinear nature of epistatic interactions limits the use of popular linear models. Several nonlinear interaction patterns are illustrated in Chapter 2, and the efficient and effective learning of discriminative epistatic interactions in Case-Control classification adds its value to the SNP-based feature selection.

1.4 Challenges

The efficient learning of discriminative epistatic interactions is subject to several challenges. We summarise three key aspects as follows:

1.4.1 Computational Challenges

One of the main difficulties in epistatic interaction detection is the intrinsic computational barrier. For a typical GWAS data containing n SNPs and m samples, the computational complexity of examining all pairs of SNPs is $mn(n-1)/2$. Currently, the SNPs can be genotyped at the level of millions. If one million SNP chip is used, total half a trillion SNP-pairs are to be tested. As been shown later in the Chapter 3 that it may take weeks, months even years in a single run for many classical methods. To speed up the data processing, researchers proposed various statistical tests, usually accompanying with a highly optimised computer code for each method using a specific hardware configuration. For higher order interactions, such as trivariate interactions, the number of tests is $mn(n-1)(n-2)/6$ and consequently the exhaustive analysis is considered computationally prohibitive [106]. Elegant works have been proposed but mainly applicable to small datasets [36] or rely on large-scale computational infrastructure [58, 59] which is not always accessible to many researchers as a routine basis.

1.4.2 Robustness of Signals

A single run through the data using a specific method to select a tiny fraction of top SNP interactions from trillions of candidates in an association analysis is essentially a filtering-based feature selection method. From the general perspective of stability in feature selection, a selection method can be sensitive to the perturbation of training samples [74, 130], leading to unstable interaction results [153]. The causes of instability problem are summarised in [46] and the small sample size with high-dimensional input space is regarded as the most challenging factor. An ad-hoc method may tend to be biased and overfit to a particular set of training samples, generating unstable results when variations are introduced into the training data. Such results contain false positives, reduce the confidence of domain experts and possibly lead to incorrect biological conclusions.

To reduce the false positives, re-sampling techniques such as cross-validations and bootstrap can be applied using multiple runs to select stable features. The desired results can be determined by checking whether they consistently appear across multiple samplings [3, 26, 61]. As for epistatic interactions detection, this is even more obvious since the feature search space is quadratic for binary interactions and cubic for ternary interactions when the sample size is relatively small. In summary, a feature selection method is ideally incorporated into the re-sampling protocols for selecting reliable results that are effectively against the sampling variations in training data. But this becomes less practical for epistasis detection since a single run by itself takes long processing time, many repeated re-sampling trials lead to unreasonable waiting time for research.

1.4.3 Methodology Challenges

Since there is no gold standard in defining the statistical tests for epistatic interactions, each existing method in association analysis may be preferable to some specific types of epistatic models or being restricted to a given order of interactions. In addition, current methods mainly focus on testing the significance of associations. Highly associated SNPs with significant P-values may not be effective predictors [56, 65] and may end up with many false positives [30]. As a result, SNP interactions detected by various association

tests need to be further investigated in terms of their predictive power in Case-Control classification. This raises a question: Are different types and orders of interactions can be further verified by an efficient machine learning method while the noise features are minimised at the same time? Although robust and model-free methods with sufficient flexibility are in demand to capture nonlinear interactions [88], the reality is that many potentially qualified machine learning methods are often black-boxes.

Often we are unsure whether a predictive model is trustworthy and whether the model works in a reasonable way as expected to identify desired features. For the proposed research topic, it is unknown of what types of interaction patterns are discriminative and which interactions are actually taken by the learning algorithms in model building. In more broader sense, any classification algorithm should be at least interpretable to some extent and make senses to human's understanding as a trustworthy method [77, 80, 112]. Similar effort should also be made toward the proposed research topic by proposing a more convincing model that correctly identifies the desired interaction features for classification.

1.5 Research Questions

The ultimate objective of proposed research is to provide high performance computational tools that are affordable to GWAS community to reveal discriminative interactions in Case-Control classification. The learned interactions may be useful clues for researchers to better understand the biological mechanisms behind. To achieve the objective, we define the following research questions:

- In literature, various association tests differed dramatically in processing time and hardware configurations. Many classical methods are slow and rely on limited computing facilities. How can we provide a unified platform for these methods to significantly reduce the runtime from the order of weeks/months to the order of minutes/hours using affordable hardware as a daily routine for research?
- Each association test generates a list of highly associated SNP interactions. Are these interactions robust under sampling variations and whether a given statistical

test is stable enough in producing genuine results? How can we make it computationally affordable to produce reliable features using the re-sampling procedure?

- For the detected robust SNP interactions, are their interaction patterns nonlinear in nature? Are they all good predictors for Case-Control classifications? If not, which interactions are useful features? Given the variability of interaction patterns, how do we build a better predictive model that is flexible, interpretable, stable and computationally efficient at the same time in realising these discriminative interaction features for effective disease prediction? Popular methods are introduced and compared in Chapter 2.

1.6 Overview of Proposed Filtering Framework

To answer these questions, we proposed a two-stage filtering framework consists of two major techniques for learning epistatic interaction features in Case-Control classification. The strategy is to first dramatically cut-down the runtime of an exhaustive search such that multiple re-sampling runs of exhaustive analyses become practical. With sufficient number of re-sampling trials, a list of robust SNP-SNP interactions can be determined based on their frequencies (or re-sampling consistency) across many trials. The selected robust ones are further verified in terms of their prediction capacity. Thus, it is necessary to perform a secondary filtering on top of the robust interactions to select only valid predictors for a Case-Control classification. In the end, the robust predictors are retained for a further interpretation. To learn such features, high performance computational methods are required by the framework to address the technical challenges.

In the first stage filtering, a universal GPU interface is proposed allowing multiple statistical tests to exhaustively analyse all SNP-pairs at each re-sampling trial in a very fast manner. Finally a set of robust SNP-pairs that consistently appear across different re-sampling trials can be determined for each test method. The overall computational cost of performing re-sampling trials is generally affordable by utilising very limited number of GPU accelerators. The produced list of robust SNP-interactions are subsequently analysed for selecting only predictive ones. In short, the first stage filtering served as a high

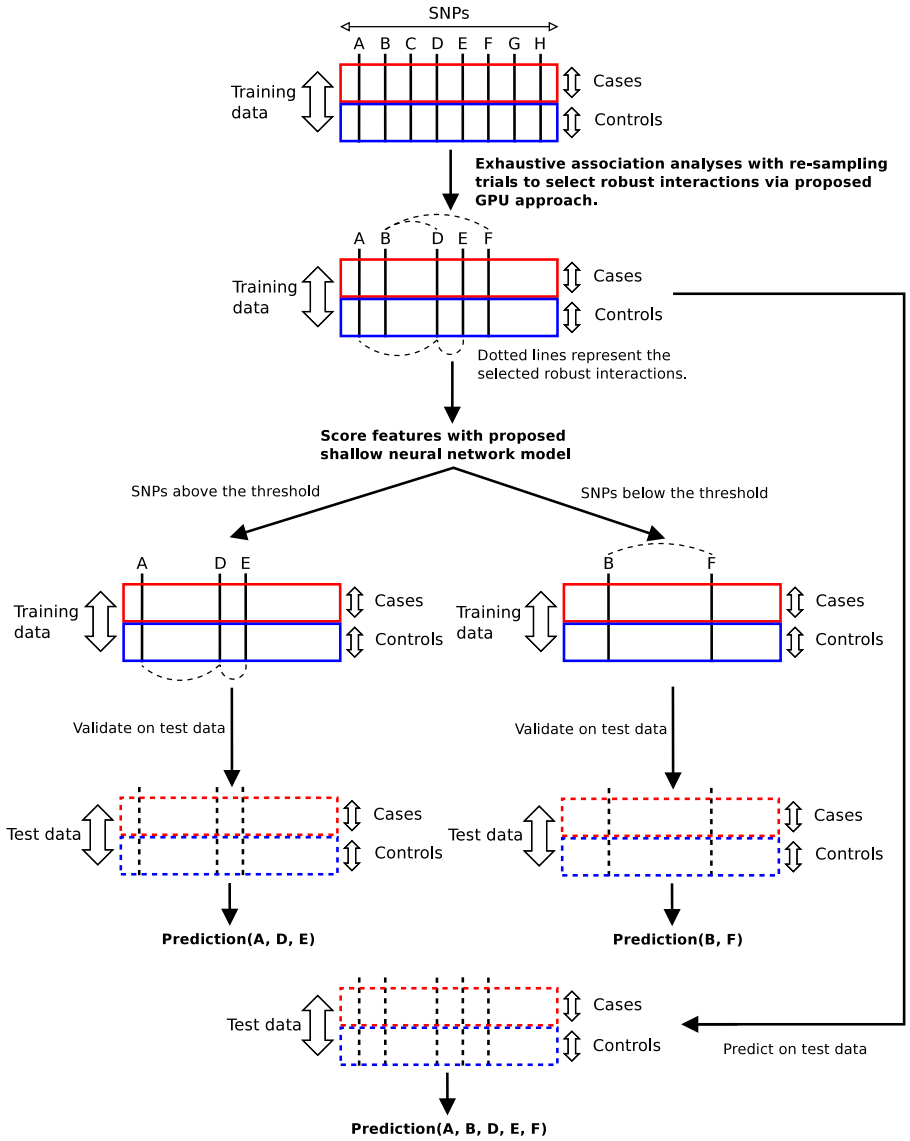

Figure 1.3: **Stage 1:** Exhaustive bivariate search is performed at each trial of re-samplings. The robust SNP-SNP interactions that commonly appear across different re-sampling trials are selected. Total 5 SNPs (ABDEF) in 4 interations are survived. **Stage 2:** The proposed shallow neural-net method scores the remaining features. Depending on the score threshold being used, remaining SNPs are separated into two groups. The performance validation for each group is performed on the test data. It turns out to be $\text{Prediction}(\text{A}, \text{D}, \text{E}) \simeq \text{Prediction}(\text{A}, \text{B}, \text{D}, \text{E}, \text{F})$, indicating that 'A-D' and 'D-E' are discriminative interactions. On the other hand, $\text{Prediction}(\text{B}, \text{F}) \simeq 0.5$ in Case-Control classification implies that the two SNPs are noise features.

performance feature selection method for robust interactions detection.

In the second stage filtering, a shallow neural network algorithm is proposed to fur-

ther examine the robust interactions; realise the discriminative nonlinear patterns within; and try to identify informative features for Case-Control classification. The signal-noise separation is effectively handled by a light-weight signal extraction procedure. The predictive model itself is computationally simple, model-free and being human understandable to some extent, and shown to be able to perform quality classifications and identify meaningful features for GWAS data. The proposed model is kept shallow for its architecture to avoid expensive costs of computation incurred by a necessary but time-consuming experiment protocol such as cross-validation. The high-level overview of the framework is depicted in Figure 1.3, applicable to different Case-Control GWAS studies.

In summary, the robustnesses and the predictive power of selected interaction features need to be both satisfied before these interactions are considered as valid signals or useful reference knowledge. Novel methods are in demand to address the large search space and the complexity of nonlinear patterns at the same time, and the proposed filtering framework responds to such requirements.

1.7 Thesis Structure and Contributions

The rest of this thesis is organised as follows:

1.7.1 Chapter 2 - Literature Review

In this chapter, classical methods that are used for epistatic interactions detection are discussed first. Based on the nature of these methods, they are categorised into two groups: association tests and machine learning family. We describe their principles, make comparisons between methods and highlight their strengths and weaknesses.

Secondly, apart from the methodology discussions, we also describe the nonlinear nature of epistatic interactions. Several nonlinear interaction patterns are illustrated to demonstrate the complexity of the task. Thirdly, the importance of the re-sampling-based stable feature selection is emphasised in this chapter. Lastly, several metrics used to assess the prediction performance of Case-Control classification are briefly introduced.

1.7.2 Chapter 3 - High Performance Learning of Robust Interactions

In this chapter, we introduce a GPU-based computing approach for learning robust SNP-SNP interactions in GWAS data. The proposed method allows customised association tests to run in a ultra-fast manner via a unified GPU interface. Significant runtime improvements are observed for the exhaustive search, making the re-sampling-based protocol feasible for selection of stable interactions.

This chapter emphasises on the selection of robust interactions using GPU accelerator. Many statistical tests varied dramatically in processing time in their original implementations, which are subject to the software implementations and optimisations, specific hardware availability, computing paradigms and the nature of the methods. Some classical methods require several weeks of time to run through a medium-sized GWAS data, leading to unreasonable waiting time. If re-sampling routines are performed directly in such a way, then the search for stable features becomes unrealistic. In this chapter, we show that the GPU-based method is fast enough such that the exhaustive search-based re-sampling trials are computationally affordable. Additionally, we address several bottlenecks, such as data loading and redundant calculations in the re-sampling procedure, and propose improved algorithms to further increase the computational efficiency.

In the experiment, five popular filters are implemented by the GPU platform and the runtime improvement for each method is demonstrated using a large GWAS simulation dataset, while the detected robust interactions from a large number of re-sampling trials are shown for three real world GWAS studies. In summary, this chapter provides high performance technical foundation for learning robust SNP-interactions, discussed mainly from a computational point of view.

1.7.3 Chapter 4 - Learning Discriminative Signals by Shallow Model

In this chapter, we introduce a shallow neural network model for learning discriminative SNP interactions. The proposed method performs effective binary classification and complex feature learning at the same time. The model contains an input layer and an hidden layer (also served as the output layer) only. During the training, the model aims

at learning a set of weights that guide the mapping from the input to the hidden layer to form two distinct clusters for the two classes respectively. An unknown instance can be subsequently mapped to either cluster to determine the classification. After the model training, the features can be scored for their relative contributions to the classification by interpreting the learned weights.

The scored features are turn out to be capturing high-level class-specific information. By illustrating with computer vision applications, the features are intuitively human understandable and consequently agree with the human's cognition to the meaningful features. To actually separate the informative features from the noise, a simple signal extraction procedure can be applied. We demonstrate with simulated datasets of nonlinear interactions that various types of nonlinear combinations can be realised by the model, and distinguished from the noise features by the extraction procedure.

The theory and algorithms are first introduced in this chapter, followed by a number of nonlinear studies showing its nonlinear approximation ability. The proposed model is then benchmarked against a classical machine learning algorithm and proven to be a competitive classification method. With sufficiently well performance of classification, the model's feature selection ability, on the other hand, deserves a further investigation. The convincing representations of learned features, shown in this chapter, increase the confidence for the derived model. Moreover, due to the simplicity of the method, it is easily incorporated in to an expensive experiment protocol, such as cross-validation, to give a relatively reliable estimation of performancce. In summary, this chapter provides a high performance shallow model for learning high-level complex features, discussed mainly from the discriminative feature selection point of view.

1.7.4 Chapter 5 - Learning Epistatic Interaction Features in GWAS

In this chapter, we demonstrate the two-stage filtering framework for systematically learning the epistatic interaction features for Case-Control discrimination. We show in this chapter that it is indeed insufficient to solely rely on association tests for detecting such interactions. Even if the SNP interactions detected by these tests in the re-sampling trials are proven to be robust, their predictive power can not be guaranteed, and a non-

negligible amount of false positives exist in the list of robust interactions. It is also shown that different association tests favour signals from certain chromosomes in a GWAS population cohort, and hence all detected SNP interactions by different methods need to be further verified for their predictive power.

We first give an overview of the problem definition and then summarise a series of associated technical challenges. Different factors that are crucial to the overall performance of the framework are discussed. After that, the framework utilising two proposed techniques is introduced in detail. For the experiment, a multi-cohort breast cancer GWAS study is used to comprehensively test the learning capacity of the framework. To highlight the discoveries, we give prediction results for the discriminative SNP features. Apart from that, we illustrate some interaction patterns that maximise the performance discrepancy between the signals and the noise on the test cohort. The visualisation of learned discriminative interaction patterns may help to understand the nature of epistatic interactions and identify possible high-risk genotype combinations for evaluation.

1.7.5 Chapter 6 - Conclusion and Future Work

This chapter recapitulates the project and provides the conclusions reached from the analyses. The future development of the methods are suggested in the end.

1.8 List of Publications

- Qiao Wang, Fan Shi, Andrew Kowalczyk, Richard M Campbell, Benjamin Goudey, David Rawlinson, Aaron Harwood, Herman Ferra and Adam Kowalczyk, **GWISFI: A universal GPU interface for exhaustive search of pairwise interactions in case-control GWAS in minutes**, 2014 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Belfast, UK, 2014, pp. 403-409.
- Qiao Wang, Sylvia Young, Aaron Harwood and Cheng Soon Ong, **Discriminative concept learning network: Reveal high-level differential concepts from shallow architecture**, 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland, 2015, pp. 1-9.

- Qiao Wang, Aaron Harwood, Miroslaw Kapuscinski, Cheng Soon Ong and John Hopper, **Efficient learning of robust and discriminative SNP-SNP interactions for Case-Control GWAS**, *Bioinformatics*. (to be submitted)

Chapter 2
Literature Review

IN this chapter, we give details of classical association tests and machine learning methods that are popular for learning epistatic interactions. All methods are covered with theoretical foundations to describe their principle and compared in terms of their characteristics, strengths and weaknesses. Additionally, the nonlinear interaction models with representative examples are shown to facilitate an understanding of the complexity of the proposed research topic. Finally, evaluation metrics used to assess a model's prediction performance throughout the experiments of the thesis are introduced.

2.1 Overview

Epistasis detection has been a major research focus of population genetics research. A large number of statistical tests have been proposed with the main aim of associating the genotypes with the phenotypes. These methods can take different strategies such as pre-filtering search, exhaustive search, heuristic search, etc. We introduce some popular methods in detail. In addition to association-based tests, machine learning methods focus on building predictive models from GWAS data. Some of them are suitable for epistasis detection but with intrinsic limitations, such as model stability issue, others mainly focus on predictions but without a clear understanding of what kind of interactions are used in model-training. Different aspects of these methods are discussed in this chapter. We review and compare them according to different criteria.

In addition, the choice of a suitable predictive model depends on the nonlinear nature of epistatic interactions. For pairwise interactions, various interaction models are covered by pioneer investigations. We illustrate a fraction of nonlinear models to demonstrate the variability and complexity faced by a predictive model. After that, we discuss the stability issue for robust signal detection, review several re-sampling techniques and highlight

its important role in feature selection. Finally, we introduce several evaluation metrics used for measuring the classification performance.

2.2 Strategies for Epistatic Interaction Detection

In this section, both association test-based methods and machine learning models are introduced and compared in detail.

2.2.1 Pre-filtering Strategy

Since the search space of bivariate interactions is quadratic, many existing methods adopt a pre-filtering strategy, in which all SNPs are tested individually, and based on a specific threshold, only a fraction of SNPs that pass the threshold are fed into a second stage test for interaction detection. This strategy is widely used due to its simplicity and computational efficiency (computation complexity is generally $O(n)$ for n SNPs in the pre-filtering stage), and does not require access to special hardwares and computing facilities. It greatly reduces the search space from the original input dimension to a manageable small set of features on which an exhaustive search of bivariate interactions can be applied. This strategy is based on the assumption of marginal effects in each locus of a SNP-pair [12, 86], making them detectable by the pre-filtering stage.

The success of this method generally relies on a threshold applied to the pre-filtering stage. As the threshold becomes more stringent, a smaller set of SNPs are retained for the second stage test, at the risk of losing interactions with small marginal effects. If the threshold is set in the other way round, then more exhaustive tests are required at the cost of increased computation burden in the second stage screening. The trade-off between the detection power and search space is a challenging issue [144]. When interaction models are nonlinear with little main effects, the pre-filtering strategy has little power to detect such genuine interactions. This has proven to be the case for colon cancer study in which important interactions were missed [160]. Consequently, exhaustive bivariate search is preferable to two-stage filtering [29]. However, it does not automatically imply that an exhaustive search-based association test can definitely capture nonlinear

interactions with little main effects since some exhaustive methods may simply consider the associations between the genotypes and the phenotypes regardless of considering whether such associations are driven by main effects or epistatic interaction effects [94].

2.2.2 Exhaustive Search

From previous discussions, we know that interactions with zero main effects are going to be missed by a univariate-based analysis, and hence the exhaustive search is inevitable. Given this search strategy, different association tests can be proposed in theory to capture pure interaction effects. However, due to the huge search space, the test method itself should be restricted by a set of finite light-weight operations to measure such effects for every possible interaction in actual computation.

In this research field, a significant effort has been made to devise new methods for learning epistatic interaction effects. We introduce several popular ones that adopted the exhaustive search strategy in this section.

Pearson's Chi-squared Test

The Chi-squared test χ^2 [73, 161] is a frequently used goodness-of-fit test, and usually used in the exhaustive search [158]. The test checks whether the genotype combinations of the two SNPs are independent from the phenotype (Case and Control) under the null hypothesis that no significant associations can be observed. If the genotypes are independent from the phenotype, then the ratio of cases to controls of a given compartment in a 3×3 contingency table (Table 2.1) should be equal to the ratio of total cases to total controls. Given the total number of samples for a genotype combination and the case-control ratio, χ^2 test can measure, for that combination, a deviation of an observed genotype frequency from the expected frequency for each of the Case and Control groups.

The χ^2 statistic that is used for testing the association between a SNP-pair and the phenotype usually take the following form:

$$\chi^2 = \sum_{i=0}^{2} \sum_{j=0}^{2} \sum_{c=0}^{1} \frac{(e_{ij}^c - n_{ij}^c)^2}{e_{ij}^c},$$

Table 2.1: A 3×3 contingency table shows frequencies of 9 possible genotype combinations. Here, n represents the sample counts and the superscript $c \in \{1,0\}$ represents the class, either Case or Control.

Genotype	$j = 0$	$j = 1$	$j = 2$	Total
$i = 0$	n^c_{00}	n^c_{01}	n^c_{02}	n^c_{0*}
$i = 1$	n^c_{10}	n^c_{11}	n^c_{12}	n^c_{1*}
$i = 2$	n^c_{20}	n^c_{21}	n^c_{22}	n^c_{2*}
Total	n^c_{*0}	n^c_{*1}	n^c_{*2}	n^c_{**}

where

$$e^c_{ij} = n^*_{ij} \times \frac{n^c_{**}}{n^*_{**}}. \tag{2.1}$$

The statistic is asymptotically a χ^2 distribution. To reject the null hypothesis, a large value is required but the strength of the association requires additional measurements to tell [91, 161]. In theory, χ^2 test is easily extensible to any higher order interactions as the dimensionality of contingency table increases. For example, the following equation can be used for three way interactions based on a $3 \times 3 \times 3$ contingency table:

$$\chi^2 = \sum_{i=0}^{2} \sum_{j=0}^{2} \sum_{k=0}^{2} \sum_{c=0}^{1} \frac{(e^c_{ijk} - n^c_{ijk})^2}{e^c_{ijk}},$$

where

$$e^c_{ijk} = n^*_{ijk} \times \frac{n^c_{***}}{n^*_{***}}. \tag{2.2}$$

In reality, the test starts losing its power as the order of interactions increases since the limited number of samples are distributed to the growing number of compartments in the contingency table, triggering the curse of dimensionality problem. As a rule of thumb, all numbers in the compartments are expected to be greater than 5 for a reliable inference [90]. Nevertheless, it is widely used in numerous studies as a primary statistic to examine pairwise interactions [19, 159, 160]. However, the χ^2 test had been considered not explicitly taking interactions into account and possibly leading to pairwise interactions dominated by strong main effects [8].

Odds Ratio

The term odds is commonly defined as the ratio of the probability of an event occurring $P(E)$ to the probability of not occurring $P(\overline{E})$: $Odds = P(E)/P(\overline{E})$. The odds ratio (OR) measures the association between two events E_1 and E_2 by comparing the odds of E_1 in the presence of E_2 and the odds of E_1 in the absence of E_2:

$$OR = \frac{P(E_1|E_2)/P(\overline{E_1}|E_2)}{P(E_1|\overline{E_2})/P(\overline{E_1}|\overline{E_2})}.$$

The $OR > 1$ and $OR < 1$ represent a positive and negative association respectively while $OR = 1$ indicates that E_1 and E_2 are independent from each other.

It is frequently used to measure whether a medical outcome is associated with an exposure (or treatment). In the context of Case-Control GWAS, the interest is to see whether the trait of interest is associated with a particular allele, genotype or genotype combination. Such connection is usually represented by a 2×2 table (Table 2.2).

Table 2.2: A 2×2 table. The frequencies in the presence and absence of a particular genotype combination are filled in the table for two conditions.

	Case	Control	Total
g_{ij}	n_{ij}^1	n_{ij}^0	n_{ij}^*
$g_{\overline{ij}}$	$n_{\overline{ij}}^1$	$n_{\overline{ij}}^0$	$n_{\overline{ij}}^*$
Total	n_{**}^1	n_{**}^0	n_{**}^*

For a pair of SNPs with total 9 possible genotype combinations, we consider a particular genotype combination g_{ij}. The odds, given that combination, is $Odds(g_{ij}) = n_{ij}^1/n_{ij}^0$, and the odds in the absence of that combination is $Odds(g_{\overline{ij}}) = n_{\overline{ij}}^1/n_{\overline{ij}}^0$. Therefore $OR = Odds(g_{ij})/Odds(g_{\overline{ij}}) = (n_{ij}^1 n_{\overline{ij}}^0)/(n_{ij}^0 n_{\overline{ij}}^1)$. This genotype combination can be regarded as a deleterious factor for $OR > 1$ or considered as a protective factor for $OR < 1$. The conclusion of no association with the phenotype can be made when $OR = 1$.

Its confidence interval (CI) can be simply derived by [1,103]. Firstly, when the sample size is not large enough the sampling distribution of OR statistic is easily skewed but the $log(OR)$ asymptotically follows the normal distribution: $log(\hat{OR}) \sim N(log(OR), \sigma^2_{log(OR)})$.

The standard error SE is defined as:

$$SE_{log(\hat{OR})} = \sqrt{1/n_{ij}^{1} + 1/n_{ij}^{0} + 1/n_{\overline{ij}}^{1} + 1/n_{\overline{ij}}^{0}}. \tag{2.3}$$

Given the significance level at $\alpha = 0.05$, one can construct a $100(1-\alpha)\% = 95\%$ CI as follows:

$$95\%CI_{log(\hat{OR})} = log(\hat{OR}) \pm z_{\alpha/2}\, SE_{log(\hat{OR})}, \tag{2.4}$$

where the critical value z depends on the adopted confidence level. For a 95% confidence level, the corresponding $z = 1.96$ in this case. Once the $CI_{log(\hat{OR})}$ is determined, the $CI_{\hat{OR}}$ can be simply calculated by taking an exponential of the lower and upper bound respectively and end up with the following interval:

$$95\%CI_{\hat{OR}} = [e^{log(\hat{OR}) - z_{\alpha/2}\, SE_{log(\hat{OR})}}, e^{log(\hat{OR}) + z_{\alpha/2}\, SE_{log(\hat{OR})}}]. \tag{2.5}$$

Additionally, the null hypothesis of $H_0 : \hat{OR} = 1$ for no association equals to $H_0 : log(\hat{OR}) = 0$. A Z-statistic which follows a normal distribution is then given by $Z = (log(\hat{OR}) - 0)/SE_{log(\hat{OR})}$ to derive the significance value for hypothesis test. This idea was applied in an elegant work describing a genome-wide SNP-SNP interaction scan called SHEsisEpi [54]. It exhaustively scans all pairs of SNPs to detect risk epistasis [5]. For each SNP-pair the algorithm calculates an OR for each of the 9 possible genotype combinations and check whether any one of them significantly deviates from 1. The P-value of the best genotype combination is obtained from the Z-statistic. The work comes with a GPU implementation, making it practical for genome-wide scanning. Lastly, when higher order interactions are considered, more genotype combinations appeared as the increased dimension of contingency table. For any combination, a 2×2 table can be created for constructing an OR statistic, and SHEsisEpi is extensible to higher order interactions in theory. Another OR-based method is called FastEpistasis, part of the PLINK software package [109, 110]. This work uses the following equation to measure the strength of the interaction between two SNPs:

$$Z = (log(OR^{1}) - log(OR^{0}))/\sqrt{(SE^{1})^{2} + (SE^{0})^{2}}, \tag{2.6}$$

where the OR^c for $c \in \{1,0\}$ and standard error are estimated as

$$OR^c = (\alpha^c_{11} \cdot \alpha^c_{22})/(\alpha^c_{12} \cdot \alpha^c_{21}), \tag{2.7}$$

$$SE^c = \sqrt{1/\alpha^c_{11} + 1/\alpha^c_{12} + 1/\alpha^c_{21} + 1/\alpha^c_{22}} \tag{2.8}$$

from the 2×2 table $[\alpha^c_{ij}]_{1 \leq i,j \leq 2}$ into which the original 3×3 table $[n^c_{ij}]_{0 \leq i,j \leq 2}$ is collapsed:

$$\alpha^c_{11} = 4n^c_{00} + 2n^c_{01} + 2n^c_{10} + n^c_{11}, \tag{2.9}$$

$$\alpha^c_{21} = 4n^c_{20} + 2n^c_{21} + 2n^c_{10} + n^c_{11}, \tag{2.10}$$

$$\alpha^c_{12} = 4n^c_{02} + 2n^c_{01} + 2n^c_{12} + n^c_{11}, \tag{2.11}$$

$$\alpha^c_{22} = 4n^c_{22} + 2n^c_{21} + 2n^c_{12} + n^c_{11}. \tag{2.12}$$

Due to the popularity of FastEpistasis, it is used in later chapters for learning robust SNP-SNP interactions.

Pearson's Correlation Coefficient

Pearson's correlation coefficient is another popular statistic measuring the strength of association between two variables. In the context of epistatic interactions detection, it can be used to measure the genotypes of two interacting SNPs in a population group:

$$\rho_{SNP_A, SNP_B} = \frac{cov(SNP_A, SNP_B)}{\sigma_{SNP_A} \sigma_{SNP_B}},$$

where cov is the covariance of the two variables

$$cov(SNP_A, SNP_B) = E[(SNP_A - E(SNP_A))(SNP_B - E(SNP_B))],$$

and σ_{SNP} is the standard deviation of that SNP. Its value ranges between -1 and 1, where -1 and 1 indicate the negative and positive correlations respectively, and 0 represents the independence between two variables.

A representative work called EPIBLASTER [62] adopted this statistic in the exhaustive filtering stage. A difference in Pearson's correlation coefficients (DPCC) between the two categories (Case and Control) is calculated for all possible interactions:

$$DPCC := \rho^1_{A,B} - \rho^0_{A,B} = \sum_{x=1}^{n^1_{**}} \frac{(g^1_A(x) - \overline{g^1_A})(g^1_B(x) - \overline{g^1_B})}{(n^1_{**}-1)\sigma_{g^1_A}\sigma_{g^1_B}} - \sum_{x=1}^{n^0_{**}} \frac{(g^0_A(x) - \overline{g^0_A})(g^0_B(x) - \overline{g^0_B})}{(n^0_{**}-1)\sigma_{g^0_A}\sigma_{g^0_B}},$$

where $\sigma_{g^c_s} = \sqrt{\sum_{x=1}^{n^c_{**}}(g^c_s(x) - \overline{g^c_s})^2/(n^c_{**}-1)}$ and $g^c_s(x)$ represents a genotype $g \in \{0,1,2\}$ of a SNP $s \in \{A,B\}$ for an instance x, belonging to a class $c \in \{1,0\}$. The correlation coefficient $\rho^c_{A,B}$ can be further decomposed into contingency table counts as follows:

$$\rho^c_{A,B} = \frac{\sum_{x=1}^{n^c_{**}}(g^c_A(x)g^c_B(x) + \overline{g^c_A}\,\overline{g^c_B} - g^c_A(x)\overline{g^c_B} - g^c_B(x)\overline{g^c_A})}{(n^c_{**}-1)\sigma_{g^c_A}\sigma_{g^c_B}} = \frac{\sum_{i=0}^{2}\sum_{j=0}^{2} n^c_{ij}\,i\,j + n^c_{**}\,\overline{g^c_A}\,\overline{g^c_B} - \overline{g^c_B}\,\sum_{i=0}^{2} n^c_{i*}\,i - \overline{g^c_A}\,\sum_{j=0}^{2} n^c_{*j}\,j}{(n^c_{**}-1)\sigma_{g^c_A}\sigma_{g^c_B}}, \quad (2.13)$$

where

$$\overline{g^c_A} = \frac{1}{n^c_{**}}\sum_{i=0}^{2} n^c_{i*}\,i, \qquad \overline{g^c_B} = \frac{1}{n^c_{**}}\sum_{j=0}^{2} n^c_{*j}\,j,$$

$$\sigma_{g^c_A} = \sqrt{\frac{\sum_{i=0}^{2} n^c_{i*}\,i^2 + n^c_{**}\,(\overline{g^c_A})^2 - 2\overline{g^c_A}\,\sum_{i=0}^{2} n^c_{i*}\,i}{n^c_{**}-1}},$$

$$\sigma_{g^c_B} = \sqrt{\frac{\sum_{j=0}^{2} n^c_{*j}\,j^2 + n^c_{**}\,(\overline{g^c_B})^2 - 2\overline{g^c_B}\,\sum_{j=0}^{2} n^c_{*j}\,j}{n^c_{**}-1}}.$$

As can be seen from equation 2.13, all components can be represented by contingency table counts. The statistic is used in the first-stage filtering of EPIBLASTER to rank all SNP-pairs in an exhaustive way. The logistic regression is applied later on the survived SNP-pairs to assign P-values to individual effects and interaction terms. The exhaustive first-stage filtering is the major computation bottleneck, and the authors developed the a GPU implementation to speed-up the processing. Throughout this thesis, the name EPIBLASTER refers to its first-stage filtering only.

Information Gain

The Shannon entropy H measures the uncertainty of a given variable. The higher the H, the more uncertainty of the target variable. Information gain (IG) is about the reduction of the uncertainty of the target variable given additional variables. To characterise interactions, IG method is proposed using information theory [27, 52, 53]. According to this idea, the genotypes at loci A and B can be treated as categorical random variables X_A and X_B respectively, while denote Y to the phenotype variable. If a bivariate interaction is considered, then IG can be calculated by first calculating the mutual information between the phenotype variable Y and two genotype variables X_A and X_B jointly; and then calculating the mutual information between Y and each genotype variable individually; and finally calculating the difference between the two mutual information. Therefore, the information gain (IG) of Y given the knowledge of genotypes X_A and X_B is given in terms of the mutual information (MI):

$$IG(X_A, X_B; Y) = MI(X_A, X_B; Y) - MI(X_A; Y) - MI(X_B; Y). \tag{2.14}$$

This formula subtracts the individual main effects of each SNP from their joint effect to estimate a gain in information. The mutual information between two discrete random variables $MI(X;Y) = H(Y) - H(Y|X)$ represents the reduction in uncertainty in Y given knowledge of X, where H is the information entropy $H(X) = -\sum_x P(X = x)\log P(X = x)$ or the amount of information required to describe the random variable X. The conditional entropy can be written in terms of the entropy of the joint distribution of X and Y as $H(Y|X) = H(X,Y) - H(X)$. The details of IG calculation for equation 2.14 can be expressed as follows:

$$MI(X_A, X_B; Y) = H(Y) + H(X_A, X_B) - H(X_A, X_B; Y), \tag{2.15}$$

$$MI(X_A; Y) = H(Y) + H(X_A) - H(X_A; Y), \tag{2.16}$$

$$MI(X_B; Y) = H(Y) + H(X_B) - H(X_B; Y), \tag{2.17}$$

$$H(Y) = -(\frac{n^1_{**}}{n^*_{**}} log \frac{n^1_{**}}{n^*_{**}} + \frac{n^0_{**}}{n^*_{**}} log \frac{n^0_{**}}{n^*_{**}}), \tag{2.18}$$

$$H(X_A, X_B) = -\sum_{i=0}^{2}\sum_{j=0}^{2}\frac{n_{ij}^1 + n_{ij}^0}{n_{**}^*}\, log\frac{n_{ij}^1 + n_{ij}^0}{n_{**}^*}, \tag{2.19}$$

$$H(X_A) = -\sum_{i=0}^{2}\frac{n_{i*}^1 + n_{i*}^0}{n_{**}^*}\, log\frac{n_{i*}^1 + n_{i*}^0}{n_{**}^*}, \tag{2.20}$$

$$H(X_B) = -\sum_{j=0}^{2}\frac{n_{*j}^1 + n_{*j}^0}{n_{**}^*}\, log\frac{n_{*j}^1 + n_{*j}^0}{n_{**}^*}, \tag{2.21}$$

$$H(X_A; Y) = -\sum_{c=0}^{1}\sum_{i=0}^{2}\frac{n_{i*}^c}{n_{**}^*}\, log\frac{n_{i*}^c}{n_{**}^*}, \tag{2.22}$$

$$H(X_B; Y) = -\sum_{c=0}^{1}\sum_{j=0}^{2}\frac{n_{*j}^c}{n_{**}^*}\, log\frac{n_{*j}^c}{n_{**}^*}, \tag{2.23}$$

$$H(X_A, X_B; Y) = -\sum_{c=0}^{1}\sum_{i=0}^{2}\sum_{j=0}^{2}\frac{n_{ij}^c}{n_{**}^*}\, log\frac{n_{ij}^c}{n_{**}^*}. \tag{2.24}$$

Since the main effects are removed from the joint effects, IG method well represents the interaction effects. A positive value indicates the synergistic effects while a negative one tells the redundancy of the variables [53].

Logistic Regression

Logistic regression is a widely used method to model the binary outcome using the log odds of the probability of an event in the form of the linear combination of a series of weighted predictors. Effort has been made to extend it to epistatic interaction detection in Case-Control studies. The general idea is to test the departure from the pure additive model. As been shown previously, the full model is given by equation 1.2. A relatively simple additive model can be derived by removing the regression coefficient β_3 which represents the interaction term.

With the likelihoods (L_f and L_a for full and additive models respectively) of the two models estimated at their respective maximum likelihood estimations, a likelihood ratio test can be used in theory to compare $log(L_f)$ and $log(L_a)$ [141] to determine whether there is a significant improvement with the inclusion of the interaction terms.

To formulate the discussion, we adopt a frequently used dummy variable coding that

represents different genotypes for the two SNPs A and B [13,23]. The idea is to first take the additive model as:

$$log(\frac{P(D)}{1-P(D)}) = \alpha + \beta_1 g_{A=1} + \beta_2 g_{A=2} + \beta_3 g_{B=1} + \beta_4 g_{B=2}, \tag{2.25}$$

where g_A and g_B are dummy variables. The four β values are regression coefficients representing the additive effects.

On the other hand, the full model is defined as follows:

$$log(\frac{P(D)}{1-P(D)}) = \alpha + \beta_1 g_{A=1} + \beta_2 g_{A=2} + \beta_3 g_{B=1} + \beta_4 g_{B=2} + \\ \beta_5 g_{A=1} g_{B=1} + \beta_6 g_{A=1} g_{B=2} + \beta_7 g_{A=2} g_{B=1} + \beta_8 g_{A=2} g_{B=2}. \tag{2.26}$$

Next, a null hypothesis of no interaction effects is constructed as:

$$H_0 : \beta_5 = \beta_6 = \beta_7 = \beta_8 = 0.$$

To test whether the four interactions terms are negligible, the likelihood ratio statistic (LRS) that measures the difference between the two log-likelihoods can be given in the following form [1]:

$$LRS = -2log(\frac{L_a}{L_f}) = 2(log(L_f) - log(L_a)).$$

A significant departure from the additive model indicating the multiplicative effects for the interaction, while other possibilities of more complicated patterns than a multiplicative model are suggested if null hypothesis is not rejected [92].

Logistic regression is a parametric method and is not model-free. The method is limited to lower order interactions detection due to the curse of dimensionality problem. The saturated full model can be unstable and may overfit to the data due to many parameters are estimated [13], and the parameter estimation can be expensive [106]. Also, the logistic regression-based tests favour the detection of multiplicative effects. In other words, it is sensitive in detecting the Logical AND model [13], which is introduced by

[75]. For other nonlinear interaction models, a more flexible method may be required. Nevertheless, this method is widely used in many studies with fast GPU implementation available for genome-wide scanning [155].

2.2.3 Machine Learning Methods

In this section, we introduce popular machine learning methods that are useful for epistatic interactions detection.

Multifactor Dimensionality Reduction

Multifactor dimensionality reduction (MDR) [115] is a classical method for modeling high-order nonlinear SNP-interactions [134], and regarded as a reference method in epistasis detection [15]. We know from previous discussions that, many existing methods (e.g. χ^2, OR, logistic regression, etc) suffer from the curse of dimensionality problem. The contingency table counts may be very sparse for higher order interactions. Due to the relatively small sample size, some compartments of contingency table may be empty. The sparsity in genotype combinations may lead to unreliable estimation (e.g. logistic regression [51, 115]).

In response, the original MDR method is proposed to reduce from high dimension space of genotype combinations to one dimension by pooling genotype combinations into either high or low risk group for association with disease risk. This method is non-parametric and model free, hence it does not require any parameter estimation and does not assume any genetic model in the first place. MDR is designed to detect epistatic interactions without significant main effects [40].

The general idea of MDR method is the following: From a pool of candidate SNPs, MDR considers n-order models. For a bivariate interaction where $n = 2$, a Case-to-Control ratio is calculated for each cell of the 3×3 contingency table (2-dimensional space). Based on a pre-defined threshold, each compartment is either marked as 'high-risk' if the ratio is above the threshold, or 'low-risk' otherwise. After that, a new binary variable is created by pooling all 'high-risk' cells into one group and leaving all 'low-risk'

ones to the other group. The generation of this new binary variable from multiple variables is an instance of constructive induction (CI) [93]. By using CI, the n-dimensional space is converted into 1-dimension. The relationship between the constructed variables and the phenotype is modeled by a simple probabilistic classifier in MDR [40] for evaluation of classification performance.

MDR considers all n-order models and determines the best one via 10-fold cross-validation (for each trial, the model is derived from 9/10 of samples and then predict on the remaining 1/10 for estimating the prediction error). The best model is selected based on two criteria:

- The average predictive power during the cross-validation. In the original work [115], the 10-fold cross-validation is repeated 10 times to reduce the variance of data sampling for more stable estimation of performance.
- The cross-validation consistency among multiple folds. The true signal should be robust enough against data sampling during the cross-validation, and the desired model should be consistently appearing in multiple folds (examining the number of times the same model appeared in each of the 10 training sets).

Once the best model is determined, the statistical significance is determined as follows. First, the null hypothesis is defined as no significant associations can be observed with the phenotype. Permutation tests of 1000 times are performed to construct a distribution of average cross-validation consistency. The observed average consistency of the best model is then checked to see whether it falls into the critical region of the constructed distribution, and the null hypothesis is rejected at upper-tail significant level 0.05 [115]. MDR is widely recognised in GWAS research community and its original work has the following advantages:

- Model free. MDR doesn't assume any inheritance model in detection. For complex diseases, the genetic models are still largely unknown. Thus, MDR expands the scope of detectable diseases.
- Non-parametric. Unlike parametric methods, such as logistic regression, there is no need for MDR to pre-determine the suitable number of estimated parameters. Given a dataset with a particular dimension, it is challenging to fit a model with

correct parameters. In contrast, MDR has the obvious advantage and does not require fine-tuning the parameters.

- Greatly reduced the dimensionality by grouping genotype combinations into binary variable, and here by simplified the detection process.
- Reduced type 1 error. A test method can be sensitive to the underlying data. The variants may be discovered due to chance, and a tailored test method can detect significantly-associated variants but turn out to be false positives when validating on another dataset. MDR, on the other hand, uses cross-validations in conjunction with the permutation test to select optimal models that are predictive and robust across different sampling trials. As it is shown in this thesis, this is also our focus to use re-sampling-based strategy to filter out unstable features.

On the other hand, the disadvantages of MDR are the following:

- MDR is computationally intensive. The exhaustive analysis using MDR is difficult to apply to a GWAS dataset containing too much SNPs [22, 106]. Although hardware acceleration techniques, such as GPU-based method [38] has been developed, the expensive permutation tests is still a computation bottleneck to data analysis. This is more obvious when it is used to deal with higher order interactions.
- Overly simplified binary variable (high/low risk status only) gives little information about the magnitude of disease risk [136]. Also, when the Case-to-Control ratio of a genotype combination is close to the overall Case-to-Control ratio, false positives and false negatives may arise, and this can be addressed by an odds ratio-based risk quantification method [21]. Instead of allocating a binary status to each genotype combination, this method calculates an odds ratio for each compartment of the contingency table to measure the strength of the association. Here, a large odds ratio (> 1) and a small odds ratio (< 1) represent positive and negative associations respectively, while the ratio that equals to one indicates no association. In comparison, another method uses Fisher's Exact test to select only statistically significant genotype combinations in model induction [39]. To achieve that, each genotype combination is evaluated by a Fisher's Exact test and then classified into either 'high-risk', 'low-risk' or 'unknown-risk' (Case-Control ratio of a genotype

combination is equal or close to the ratio in the whole dataset) group based on the P-value. Those labelled as 'unknown-risk' are removed from the MDR analysis, leading to more robust final model. This method works well for small sample data.

- It is hard to summarise and explain the resulting genotype combinations [115]: The derived high and low risk genotype combinations may be distributed across the contingency table, without showing a directly understandable pattern and trend. Given a particular risk group, one may expect to see some genotype combinations exhibiting interpretable information to foster the understanding of the underlying mechanism of the disease. Unlike SNPs with strong marginal effects, the interpretation to epistatic interactions, in some cases, is considered difficult in the original work. In fact, this is a shared challenge for other detection methods as well.

Kernel Methods

As a maximum margin classifier, support vector machine (SVM) has been widely-used in machine learning applications. In areas such as bioinformatics, SVM reports state-of-the-art results in many studies. For binary classification, SVM attempts to find a hyperplane that maximises margins between the two training groups. Training instances that are on the margin are called *support vectors*, whose weights decided the decision function. Based on the decision function, instances apart from the margin are classified into their respective classes accordingly. The basic idea of SVM is summarised as follows [11, 24]:

Given a set of training data $\mathcal{D} = \{\mathbf{x}_n, y_n\}_{n=1}^{N}$, we aim at learning a weight vector $\mathbf{w}$ and bias b, and classify an instance $\mathbf{x}_n$ by:

$$\mathbf{w}^T\mathbf{x}_n + b \geq +1 \text{ for } y_n = +1 \text{ if } \mathbf{x}_n \in \text{class A},$$

$$\mathbf{w}^T\mathbf{x}_n + b \leq -1 \text{ for } y_n = -1 \text{ if } \mathbf{x}_n \in \text{class B}.$$

For an arbitrary instance $\mathbf{x}_n$, its distance D to the hyperplane $\mathbf{w}^T\mathbf{x}_n + b = 0$ is

$$D = \frac{|\mathbf{w}^T\mathbf{x}_n + b|}{||\mathbf{w}||} = \frac{y_n(\mathbf{w}^T\mathbf{x}_n + b)}{||\mathbf{w}||} \geq M, \tag{2.27}$$

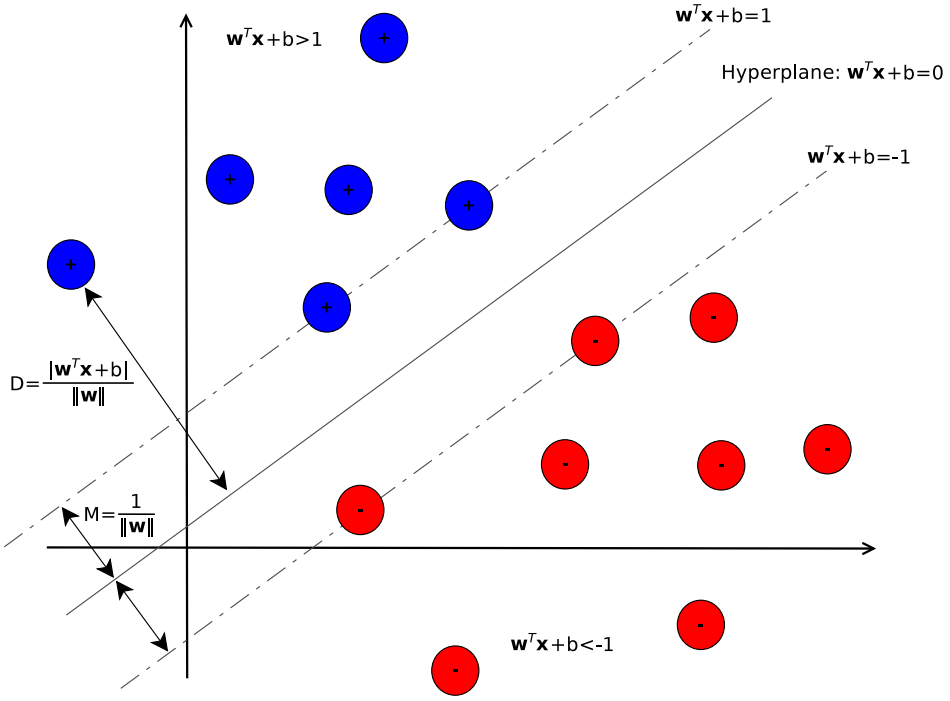

Figure 2.1: Support vector machine.

where we wish to maximise the margin M. Here, we fix $||\mathbf{w}|| \cdot M = 1$, thus maximising M is converted to minimising $||\mathbf{w}||$:

$$\arg\min_{\mathbf{w}^*} \frac{1}{2}||\mathbf{w}||^2, \text{ subject to condition } y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1. \tag{2.28}$$

To solve the solution $\mathbf{w}^*$, *Lagrange multiplies* α_n are added to form the following equation:

$$\mathcal{L}(\mathbf{w}, b, \alpha_n) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{n=1}^{N} \alpha_n(y_n(\mathbf{w}^T\mathbf{x}_n + b) - 1), \tag{2.29}$$

which received the solution at the saddle point, where the minimum is solved with respect to $\mathbf{w}$ and b, and the maximum is solved with respect to α_n. At the saddle point, the gradients are set to zero with respect to $\mathbf{w}$ and b, and we have:

$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \alpha_n)}{\partial \mathbf{w}} = \mathbf{w} - \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n = 0, \tag{2.30}$$

$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \alpha_n)}{\partial b} = \sum_{n=1}^{N} \alpha_n y_n = 0. \tag{2.31}$$

Substitute equations 2.30 and 2.31 into equation 2.29 we end up with the following:

$$\mathcal{L}(\mathbf{w}, b, \alpha_n) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n_1=1}^{N} \sum_{n_2=1}^{N} \alpha_{n_1} \alpha_{n_2} y_{n_1} y_{n_2} \langle \mathbf{x}_{n_1}, \mathbf{x}_{n_2} \rangle, \text{ subject to condition } \alpha_n \geq 0. \tag{2.32}$$

Now the function is no longer relying on $\mathbf{w}$. By maximising this quadratic function with respect to α_n here, the resulting vector $\boldsymbol{\alpha}$ is sparse with most elements α_n equal to 0. The remaining non-zero α_n constitute the final solution $\mathbf{w}^*$ by equation 2.30. The instances x_n that are corresponding to non-zero α_n are supporting vectors lying on the margin, with $y_n(\mathbf{w}^T \mathbf{x}_n + b) = 1$. Consequently, the distance of any supporting vector to the hyperplane is $1/||\mathbf{w}||$, shown in Figure 2.1.

From current introduction, SVM is linear in nature. To deal with nonlinear problems, one method is to map from original input space nonlinearly into a feature space, in which the linear SVM can apply. To simplify the discussion, much of the mathematical details are omitted here, and we directly give the general form of the corresponding quadratic function for this scenario as follows:

$$\mathcal{L}(\mathbf{w}, b, \alpha_n) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n_1=1}^{N} \sum_{n_2=1}^{N} \alpha_{n_1} \alpha_{n_2} y_{n_1} y_{n_2} \langle \mathbf{\Phi}(\mathbf{x}_{n_1}), \mathbf{\Phi}(\mathbf{x}_{n_2}) \rangle. \tag{2.33}$$

Comparing this function to equation 2.32, we can see that the dot product of two vectors in the input space is replaced by the dot product of two vectors mapped by a nonlinear function $\mathbf{\Phi}$. Generally, the dimension of feature space, defined by $\mathbf{\Phi}$, is much greater than the dimension of the original input space. Therefore, explicit mapping in such a way is costly. To address this issue, the kernel function is used to avoid such explicit mapping and the corresponding dot product calculations. A kernel function $k(\mathbf{x}_{n_1}, \mathbf{x}_{n_2}) = \langle \mathbf{\Phi}(\mathbf{x}_{n_1}), \mathbf{\Phi}(\mathbf{x}_{n_2}) \rangle$ is then used to replace the dot product term of two nonlinear mappings in equation 2.33, and hence greatly reduce the computations. The SVM's nonlinear approximation ability relies on the choice of suitable kernel(s) and fine-tuning the associated parameters. For example, a polynomial kernel $k(\mathbf{x}_{n_1}, \mathbf{x}_{n_2}) = (\langle \mathbf{x}_{n_1}, \mathbf{x}_{n_2} \rangle + 1)^d$ requires its degree of polynomial d to be properly determined. A kernel is considered valid as long as

it satisfies the Mercer's condition. Kernels are application dependent, and various kernels [10] are potentially useful. The selection of appropriate ones including the combinations of them require extra research and engineering effort to maximise the performance.

Another popular kernel method is based on the Fisher's linear discriminant analysis (FLDA). The basic idea of FLDA is to look for a projection of samples to a space, where the within-class variances are minimised while the difference of means between the two classes is maximised.

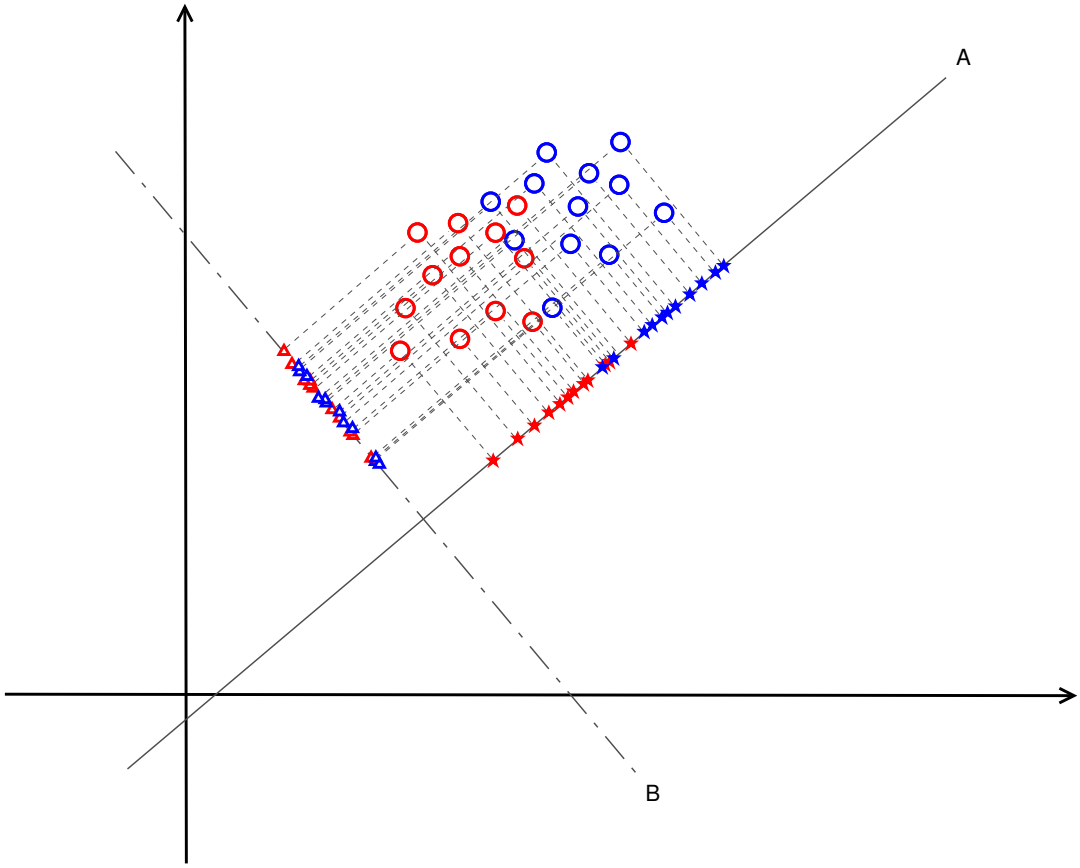

Figure 2.2: Fisher's linear discriminant analysis.

In other words, the objective is to find a projection such that, the overlaps between the two classes are minimised with greatest difference between the means of the two classes. At the same time the scatter of the instances within the same class is minimised. As can be seen from Figures 2.2, two projection lines A and B are illustrated. Among infinite possible projections for a binary classification, the projection to A has larger between-class

separations and smaller within-class variances than the projection to B. Once the projection is learned, binary classification can be simply determined by mapping the unknown instances to the well-separated clusters. The Fisher's criterion is formally described as a function of $\mathbf{w}$ and derived as follows [10]:

$$\mathcal{J}(\mathbf{w}) = \frac{(m_2(z) - m_1(z))^2}{s_1^2(z) + s_2^2(z)}, \tag{2.34}$$

where m_1 and m_2 are means and s_1^2 and s_2^2 are variances of the two classes in the projected space of the instance $z = \mathbf{w}^T\mathbf{x}$. Therefore, the objective is to maximise the function. Rewriting equation 2.34 gives:

$$\mathcal{J}(\mathbf{w}) = \frac{\mathbf{w}^T\mathbf{S}_B\mathbf{w}}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}}, \tag{2.35}$$

where the between-class covariance matrix $\mathbf{S}_B$ and within-class covariance matrix $\mathbf{S}_W$ are given respectively as follows:

$$\mathbf{S}_B = (m_2^{'}(\mathbf{x}) - m_1^{'}(\mathbf{x}))(m_2^{'}(\mathbf{x}) - m_1^{'}(\mathbf{x}))^T, \tag{2.36}$$

$$\mathbf{S}_W = \sum_{c=1}^{2} \sum_{n_c=1}^{N_c} (\mathbf{x}_{n_c} - m_c^{'}(\mathbf{x}))(\mathbf{x}_{n_c} - m_c^{'}(\mathbf{x}))^T, \tag{2.37}$$

where $m_c^{'}$ represents the mean vector of original input space for a given class c. Setting the derivatives equal to zero with respect to $\mathbf{w}$, the solution is finally found to be proportional to $\mathbf{S}_W^{-1}$ and $m_c^{'}$:

$$\mathbf{w} \propto \mathbf{S}_W^{-1}(m_2^{'}(\mathbf{x}) - m_1^{'}(\mathbf{x})). \tag{2.38}$$

FLDA works well for linear cases, but as can be seen from Figure 2.2, even projection to A contains overlapping area between the two classes. For more complex nonlinear scenarios, FLDA requires add-ons to adapt to those cases. In response, the idea of nonlinear mapping to the feature space $\mathcal{F}$ is suggested in pioneer work [95], in which the nonlinear mapping $\mathbf{\Phi}$ is used to replace elements in equations 2.36 and 2.37. Same as kernel SVM, the kernel trick is also applied here to avoid explicit computation of dot-product between the mapped vectors. Therefore, the kernel function is used instead, resulting in kernel Fisher's discriminant analysis (KFDA).

The state-of-the-art performance in countless applications places kernel methods in a dominant position in machine learning. For example, SVM uses recursive feature elimination (SVM-RFE) [41], a greedy iterative procedure to rank features. SVM-RFE starts with a full list of features, removes the least ranking feature (ranking criteria calculated based on the weights) at each iteration, re-trains the model and repeats the procedure. By the end of the training, a list of ranked features is obtained. The major drawback of the method is the intensive model training incurred by the iterative procedure. Although RFE is a heuristic search method in nature, which reduces the search space to some extent, the backward elimination procedure [119] is still expensive in computation due to the model training at each iteration. As for SNP-SNP interactions, SVM can be used in testing regression error difference for datasets with different number of SNP features. One SVM model can be trained on the dataset with all inputs while the other is trained on the data with two inputs excluded (a SNP-pair) [88]. In this method, the bootstrap is used in validating the null hypothesis of no significant change in regression error.

All kernel methods share a common challenge: the choice of suitable kernels that significantly affect the detection power [150]. The determination of specific types of kernels and their parameters *a priori* is application dependent. Each kernel may have its own view of data and may be biased to some characteristics. For example, a linear kernel is suitable for modelling additive effects while other kernels such as polynomial and Gaussian kernels reviewed in [97], are more suitable for interactions. Also there is an imbalance between the model-fitting and generalisation ability for different kernels. A RBF kernel has a local view of the data, where the effects of different *width* parameter σ apply to instances in close neighbour, while the polynomial kernel has a relatively global view and takes effects on scattered data points [125]. The latter one is considered generalised better on the unseen data but may require higher degree of polynomials for fitting well to the training data [125, 152]. Due to the data heterogeneity, a single kernel is not suitable for all scenarios [127]. To avoid the bias towards a particular part of data during the model-fitting, one can rely on a *hybrid* kernel strategy for merging different kernels to enable a more effective multiple kernel learning (MKL) [4]. The merging can be done in several different ways. For example in [126], one can consider the same kernel function

with different parameters (e.g. multiple RBF kernels with different width values). Different type of kernels can also be combined by a linear (weighted or unweighted sum) or nonlinear combination function (functions like multiplication, power, exponentiation, etc) [43]. The resulting composite kernel is required to take the model complexity into consideration, by using regularisations to allow sufficient sparsity in kernel coefficients [143]. In brief, the selection of kernels adds another layer of complexity to model training.

Neural Networks

Artificial Neural Network has been one of the major corner stones in machine learning history. Among many classical neural-net models, multilayer perceptrons (MLP) is a famous feed-forward network.

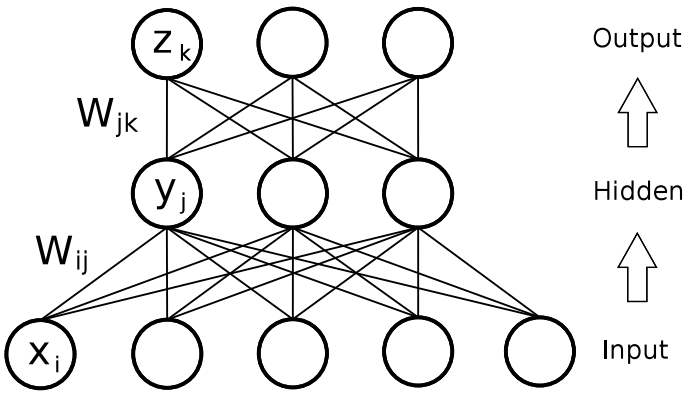

Figure 2.3: Multilayer perceptrons.

It uses parallel processing units called *neurons* to receive information from previous layer, encode and transmit to the next layer until reaching the final output layer, where the discrepancies to the presented outputs are calculated. The errors are then propagated back to adjust each layer's weights. After cyclically presenting input data to the network for sufficient training, the learned weights give a quality mapping from the input to the output, such that the discrepancies to the desired outputs are minimised (early stopping may be applied to avoid overfitting to the training data). The classification for an unseen instance in test data can then be achieved via such mapping. Due to its *universal approximation* ability [50], they are applicable to nonlinear studies and being pervasive in real

life applications. The adaptation of weights is performed by a method, widely known as *back propagation*, with the core concepts briefly introduced as follows [10,116]:

A typical MLP network consists of an input layer, an output layer and several nonlinear hidden layers in between. Given Figure 2.3, a three layer network is illustrated. The input is first feed into the network by taking a net input as $\mathbf{y} = \mathbf{W}_{ij}\mathbf{x} + b$ at the hidden layer, where each hidden neuron j then applies a nonlinear differentiable activation function $g(y_j)$ and sends out to the next hidden layer. A popular choice of the activation function is the *sigmoid* function $g(y_j) = 1/(1 + e^{-y_j})$. Finally at the output layer, a calculated output signal such as $z_k = \sum_j w_{jk} g(y_j)$, is compared against the desired result, resulting in a *squared error* loss:

$$\mathcal{E} = \frac{1}{2}\sum_k (d_k - z_k)^2, \tag{2.39}$$

where d_k is the desired output element for the output neuron k.

The loss function is then minimised with respect to the weights. Gradient descent method is usually adopted by updating the weights in the opposite direction of the gradient. For example, the update to $\mathbf{W}_{jk}$ can be expressed as:

$$\Delta w_{jk} = -\eta \frac{\partial \mathcal{E}}{\partial w_{jk}} = -\eta \frac{\partial \mathcal{E}}{\partial z_k} \cdot \frac{\partial z_k}{\partial w_{jk}}, \tag{2.40}$$

where η represents the learning rate. Denote δ_k to $-\frac{\partial \mathcal{E}}{\partial z_k}$, we have:

$$\Delta w_{jk} = \eta \delta_k g(y_j), \tag{2.41}$$

where $\delta_k = -\frac{\partial \mathcal{E}}{\partial z_k} = (d_k - z_k)$, representing the discrepancy (error) between the desired output and the actual output at the output layer. Given the calculated error δ_k for each output unit k, the error δ_j for a previous hidden layer unit j is known as the following form:

$$\delta_j = g' \sum_k \delta_k w_{jk} = g(y_j)(1 - g(y_j)) \sum_k \delta_k w_{jk}, \tag{2.42}$$

and the update to $\mathbf{W}_{ij}$ can be easily given as:

$$\Delta w_{ij} = -\eta \frac{\partial \mathcal{E}}{\partial w_{ij}} = -\eta \frac{\partial \mathcal{E}}{\partial y_j} \cdot \frac{\partial y_j}{\partial w_{ij}} = \eta \delta_j x_i. \tag{2.43}$$

By substituting equation 2.42 into 2.43, Δw_{ij} is resolved. The training is an iterative procedure. For each instance presented to the MLP, the derivation of the error term δ is initiated from the output layer and then gradually propagated back to previous hidden layers. Once they are calculated for all hidden units, the updates to the weights can be determined. With sufficient cycles of training, the loss function is minimised.

It is a common interest to investigate the coefficients that are associated with the input variables in a statistical model to measure the relative importance of the features. For example, the coefficients in a logistic regression model, the learned weights of a linear SVM model, etc., may be used in constructing feature scores. They are regarded as useful indicators in feature ranking. Such interest also applies to neural network models.

Early works investigated internal structures of the neural-nets to derive some measurements from the learned weights to quantify the relative contributions of the features. Some representative early works such as [81, 82] proposed a *Contribution Value* (CV) for scoring features, and use this method to identify useful interacting loci in early epidemiology studies [9]. However, using CV to score features was later found to be generating inconsistent results from run to run [87]. The stability issue of feature ranking in neural-net is likely due to multiple local minima in the error function [87]. The phenomenon is also verified by [60], but shown to be improved by combining with a genetic algorithm to select features as input to the network. Motivated by [81], another early work of defining feature contributions can be found in [117]. But the stability issue still exist in the later extension work [79]. Recently, several weight-based feature selection methods are evaluated in [83] for neural networks, but a global consensus of importance rankings is not satisfactorily achieved, and the stability selection remains an open question.

Another notable method called genetic programming neural network (GPNN) [114] also draw attention to the issue of inconsistent feature ranking. The traditional back propagation neural networks may get stuck into different local minima when minimising the loss function, and GPNN uses the idea of genetic programming [64] to optimise

the network architecture including the inputs, weights and the connectivity (e.g. number of nodes, hidden layers, etc) of the network. GPNN identifies useful features by evolving the architecture, and shown to have better power than back propagation networks (using a R_I scoring metric) when detecting various epistasis models in the presence of non-functional SNPs. A drawback of this method is its dependence on parallel computing resources in contrast to a single desktop machine usually required a back propagation network [114]. An extension work called grammatical evolution neural network (GENN) [104] also optimised the network architecture and further improved the efficiency and flexibility over GPNN. The main difference between the two methods is the evolutionary algorithm being adopted and GENN evolved more efficiently than GPNN in fewer generations [105]. For modeling complex patterns, it is desired to know whether a GE-optimised small neural-net can approximate a large-scale network's performance, and whether such evolutionary strategy is applicable to deep neural nets [102].

Deep learning methods, are generally considered more powerful in learning high-level patterns than shallow neural-nets. A deep learning method gradually learns more complex data representations from a lower level layer to a higher one, where the discriminative information is amplified while irrelevant variations are suppressed [69]. For low level layers, simple low-order statistical dependences can be learned and then fed into higher layer for more complex abstractions. For example, sparse coding [71, 107] and stacking-based encoder-decoder paradigm [111] can learn higher-level feature representations like edges/strokes other than the raw pixel intensities from possible computer vision applications. To learn even higher level feature representations, one can stack multiple layers of convolutional restricted Boltzmann machines into a convolutional deep belief network [72]. By illustrating the learned basis functions, this hierarchical generative model is shown to be able to learn simple edges, objects parts, until reaching the concept-level understanding. Another example is using the idea of sparse deep autoencoder to realise high-level concepts (e.g. the concept of human face) at the cost of training deep layers with thousands of computing cores [67].

Due to the high-level nonlinear approximation ability, effort has been made toward the application of deep learning models in detection of high-order SNP interactions. For

example, a stacked autoencoders method is used to extract latent representations from raw SNPs for capturing epistatic interactions [30]. In comparison, a single autoencoder is considered shallow and being very limited in learning complex representations. Although the feature extractions via multiple layers benefit the classification at the final layer, the feature selection at the input space and the interpretation to the learned model remain open questions in this work, leaving the model a black box.

To address the feature selection issue, a deep feature selection model is proposed by inserting a sparse one-to-one linear layer in between the input and first hidden layer [76], in which each hidden neuron at that layer only connects to one input feature. The weight that associated with an input feature is then used as a measurement of the feature importance. Another method called dropout feature ranking [18], based on the idea of Dropout [128] (a regularisation method that randomly disables some nodes), uses a variational dropout [34] to optimise the feature-wise dropout rate at the input space for feature ranking (small dropout rate for most important features while greater dropout rate for least important features).

Another challenge generally faced by neural network models is interpretation of the learned models. The term *interpretability* can be described in different ways [6,35,78,112], but share the same objective: enabling trust in models. In machine learning, a good prediction performance by itself may not be sufficient for model deployment. To establish a trust between the domain expert and a predictive model, it is also desired to know how and why the prediction is made, and the characteristics expressed by the model should be understandable to human [112]. In addition, the models are expected to work in a way close to human behaviours [78] to enhance the trust. Neural networks, are often criticised for being black boxes, and considered difficult to interpret due to the intrinsic nature of model learning. For deep neural networks, the robustness of interpretation is also questioned [35]: a small perturbation made to an input instance may lead to a dramatically different interpretation (e.g. different saliency maps are generated as a result of two similar inputs). This fragility reduces the confidence of domain experts in applying these models to real world applications. Lastly, the training of a deep learning model usually require to determine different-sized stacked layers; fine-tuning a large volume of

parameters; and demand long processing time with dedicated high-performance computing facilities. In summary, it is quite challenging for a neural-net model to be flexible, robust, interpretable and computationally efficient at the same time.

2.2.4 Summary of Detection Methods

So far, we have introduced popular methods for epistatic interactions detection and categorised them into two groups: association tests and machine learning methods. A brief comparison of these methods is given here.

From machine learning point of view, the detection of epistatic interactions is essentially a feature selection task. The association tests such as χ^2 test, odds ratio, Pearson's correlation coefficient-based statistic, information gain, etc., are pre-defined test statistics. With sufficient parallel computing resources, these methods are capable of exhaustively analysing all SNP interactions at various orders. The advantages of these methods are their simplicity and transparency. For most of these methods, a simple statistic can be derived explicitly from the contingency table distributions within a limited number of mathematical calculations. As been shown in Chapter 3, the overhead of calculating the statistic itself is far less than collecting contingency table counts in computation.

The limitations, on the other hand, are also obvious. Firstly, there is no gold standard definition of interactions and any potential methods require to be specifically defined. For different orders of interactions, some statistics need to be extended manually for adaptation. Besides, each statistic may be biased and only capture some types of interactions. We demonstrate later in Chapter 5 that different methods pick their favoured signals from different chromosomes. Secondly, association tests are sort of *filter* methods [55, 74, 118, 130] in feature selection, with no connection to a predictive model for a direct evaluation. The selection criteria of such method does not associate with the actual prediction capacity. Lastly, the exhaustive search using a light-weight statistic becomes computationally prohibitive for detecting high order interactions.

Machine learning methods, on the other hand, take the prediction performance estimation into consideration. A conceptually simple and effective method like MDR can detect different orders of interactions. MDR is directly applicable to exhaustive search in

theory, but the involved heavy routines such as cross-validations and permutation tests make the data processing time-consuming. In short, it is hard to apply the whole procedure in an exhaustive manner to large datasets.

Kernel methods frequently report state-of-the-art results in prediction tasks, and they can also be used in estimating the prediction capacity for a subset of selected SNP features. An initial set of features are usually presented to the model and then a pair/set of SNPs can be removed to measure the changes in performance. The least important features are then excluded and the remaining features are fed into the next iteration of selection procedure. Such iterative procedure connects the prediction evaluation to the subset selection procedure to rank features. However, due to the heavy model training, such routines are computationally more expensive than filter methods [74, 130]. In addition, the selection of proper kernels incurs extra complexity to the experiment.

Neural networks are ideal methods for modeling nonlinear interactions due to their universal approximation ability. In common with kernel methods, neural-nets are considered black boxes that make model interpretation difficult. Early research evaluated the learned weights and proposed metrics for measuring the relative importance of each feature. Some stability issues arise and genetic programming techniques can be applied to optimise the network architecture and input features. Deep neural-nets, on the other hand, are flexible for detecting higher order feature dependencies. A main focus of deep learning is feature representations by feature extractions at deep layers, with less attentions paid to feature selection at the input space. The model interpretation of deep architecture also poses some stability issues, and training such networks is very costly comparing to an ordinary shallow model.

The key strengths and weaknesses for discussed methods are summarised in Table 2.3. Each method is assessed in terms of: 1) the type of the method; 2) whether the learned interactions are also effective predictors; 3) whether it is applicable to exhaustive search; 4) whether the method is possibly favouring certain types of interaction patterns; 5) if the model is interpretable to some extent; 6) how the model is scalable to high dimensional data; and 7) major limitations.

Table 2.3: A general comparison of popular method families for epistasis detection. [♭]Odds ratio methods; [†]Pearson's correlation coefficient-based statistic; [♮]information gain methods; [‡]kernel machine-based feature selection and evaluation protocols; [§]neural networks for interactions detection. Here, we denote 'AT' and 'ML' to *association test* and *machine learning* respectively.

Methods	χ^2	OR[♭]	PCC[†]	IG[♮]	MDR	KM[‡]	NN[§]
Type	AT	AT	AT	AT	ML	ML	ML
Predictors selected	further validation required	further validation required	further validation required	further validation required	interaction models with minimum test errors	iterative procedure for interactions selection	iterative procedure or direct interpretation of weights
Exhaustive Search	yes	yes	yes	yes	in limited range	no	no
Interaction Models Dependent	possible	possible	possible	possible	model free	kernel dependent	model free
Model Interpretation	simple statistic	simple statistic	simple statistic	simple statistic	required for learned patterns	black box	black box
Scalability	difficult for high order interactions	difficult for high order interactions	difficult for high order interactions	difficult for high order interactions	scalable for small dimension	scalable for small dimension	scalable for small dimension
Main Limitation	possibly biased selection	possibly biased selection	possibly biased selection	possibly biased selection	expensive cross-validations and permutation tests	selection of correct kernels and heavy model training	stability issue and heavy model training

2.3 Stability of Interactions Detection

A very important issue in feature selection is the stability of selected features. The term *stability* is usually known as the sensitivity of a feature selection method to minor perturbation introduced into the training data [61,154], for which adding, deleting or randomly partitioning of data may end up with unstable results [74,130].

In bioinformatics applications, re-sampling procedures and multi-cohort datasets usually present different versions of training samples (of the same disease) to a feature selection method. Unstable biomarkers generated under such conditions reduce the confidence of the generalisation abilities of discovered biomarkers, and possibly impair the biological interpretations [26,61]. In contrast, stable and consistent results reduce the false positives [3]. Stable SNP-SNP interactions detection, in particular, may have a biological relationship with the trait of interest [8].

The stability of epistatic interactions may be subject to a particular association test being used or driven by the bias within the given dataset [8]. In more general sense, the characteristics of the dataset itself is considered significantly impacted on the stability selection [2]. A common practice to remedy this issue is using re-sampling protocols such as cross-validations to select stable features that robustly appear in different folds of the sampling. A typical example discussed earlier is MDR in which a cross-validation consistency is set as a hurdle for selection of robust interactions.

For exhaustive interactions detection, the stability issue becomes more critical. Given hundreds or thousands of samples, a second order exhaustive search typically requires to examine 10^{11} or more SNP pairs, pushing the dimension of feature search space to an extreme level. In such a case, even tiny variations in the training data may lead to quite different lists of selected pairs. Pursuing extremely low P-value for a calculated statistic is more likely over-fitting to a particular set of training samples in this situation. As a result, a replication/validation on another dataset may not be expected. With the support of re-sampling techniques, a large number of false positives may be filtered out by checking the re-sampling consistency of selected features.

In summary, a single run through the data is insufficient, and using a re-sampling method such as cross-validation for selecting robust interactions from various training

sets is a necessary step before follow-up analyses. However, the computational costs are generally not affordable since the runtime of a single exhaustive analysis is long enough for many existing methods, and how to speed-up the data processing is a key problem.

2.4 Nonlinear Epistatic Models

The main barrier of learning epistatic interactions is their nonlinear nature. Recall the XOR scenario from Table 1.1. In that case, each of the two SNPs is unable to independently discriminate Cases and Controls (a prediction score close to 0.5 is expected for a binary classification). Once they are presented together, an effective predictor is formed and a nonlinear model is easily employed for effective classification.

Table 2.4: Four nonlinear interaction patterns of SNP-pairs. Eight SNPs $\in \{A, \ldots, H\}$ constitute four nonlinear patterns: full-risk genotype combinations are coloured in red, while zero-risk ones are marked in blue. The ratio of counts #Cases : #Controls is given for each cell of the table.

		SNP_B		
		0	**1**	**2**
SNP_A	**0**	0 : 166	0 : 167	333 : 0
	1	0 : 167	334 : 0	0 : 167
	2	333 : 0	0 : 167	0 : 166

		SNP_D		
		0	**1**	**2**
SNP_C	**0**	0 : 76	0 : 77	153 : 0
	1	0 : 155	462 : 0	0 : 307
	2	231 : 0	0 : 385	154 : 0

		SNP_F		
		0	**1**	**2**
SNP_E	**0**	0 : 177	0 : 294	471 : 0
	1	235 : 0	59 : 0	0 : 294
	2	0 : 58	235 : 0	0 : 177

		SNP_H		
		0	**1**	**2**
SNP_G	**0**	0 : 125	0 : 125	250 : 0
	1	0 : 125	0 : 125	250 : 0
	2	250 : 0	250 : 0	0 : 500

If we examine from a data point of view, the full penetrance Table 1.1 is transformed

into Table 1.2, in which each cell only receives either Case samples or Control samples. Because of zero main effects, a univariate search strategy will miss that genuine pair and a linear predictive model will show little capacity in the classification task.

In addition to that interaction pattern, a list of full penetrance disease models of bivariate interactions (either nonlinear or linear) are reviewed and summarised in [75]. The factors such as penetrance functions, allele frequencies and other parameters, may vary greatly for real GWAS data. Therefore, the interaction patterns and cell count distributions vary in different forms.

For simplicity, we focus on nonlinear cases here and present, from the data point of view, four representative interaction patterns in Table 2.4. Using M78, M84, M85 and M114 models in [75] as an inspiration, a total 1000 samples are distributed to the compartments of either Case or Control group in the table. For the presented four interaction patterns, only nonlinear models can report quality classification results. For real GWAS datasets, main effects are not zeros in most cases and there are countless shapes of sample distribution for the table compartments. All such factors complicate the detection process and a flexible method is in demand to realise various interaction features.

2.5 Performance Evaluation Metrics

For the proposed research topic, we are interested in measuring the prediction capacity of selected features in Case-Control classifications. Because it is a typical binary classification task, the performance evaluation metrics that are widely used in machine learning research are directly applicable. In this section, we introduce two important ones that are used in the experiments throughout the thesis.

A classical metric for measuring the overall prediction performance, known as Accuracy, is given as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + FN},$$

where TP, FN, TN and FP represent true positive, false negative, true negative and false positive respectively.

Accuracy measures how many labels are correctly identified among all samples. This metric, however, is considered generating overly-optimistic bias when samples are imbalanced between the two classes and the Balanced Accuracy metric is suggested in this situation [14] as follows:

$$\text{Balanced Accuracy} = \frac{\text{Sensitivity} + \text{Specificity}}{2} = 0.5 \times \frac{TP}{TP + FN} + 0.5 \times \frac{TN}{TN + FP},$$

where

$$\text{Sensitivity} = \frac{TP}{TP + FN}, \quad \text{Specificity} = \frac{TN}{TN + FP},$$

in which the sensitivity and specificity measure how well the classification is made in each class.

Balanced accuracy takes the average of sensitivity and specificity to estimate how well the classification is made for both classes. A classifier that is highly biased towards a major class will not receive a high score using this metric. In real world classification applications, samples are normally imbalanced between the two classes, thus it is preferable to classical accuracy metric in most cases.

2.6 Summary

In this chapter, we categorised all detection methods into either association test or machine learning family. A detailed introduction is given for each method. From previous discussions, the association test-based methods are essentially filters. The computational cost is smaller than a typical machine learning method, but the detected interactions are not guaranteed to be effective predictors in classification. In addition, it is hard to scale to high order interactions. The machine learning models, on the other hand, are quite expensive in computation and are not scalable to high dimensional data. Many such methods are black boxes and often come with certain degrees of stability issues. Therefore, the stability of a feature selection method, the variability of complex nonlinear patterns and the huge search space jointly decide the great challenges faced by the research topic.

Chapter 3

High Performance Learning of Robust Interactions

IN this chapter we present a GPU-based screening platform, enabling the user-defined statistical tests to quickly analyse all SNP-SNP interactions in an exhaustive manner. Due to the fast speed, it allows robust interactions detection via re-sampling trials. The proposed GPU framework is introduced, with significant runtime improvement shown in this chapter. In addition, algorithm improvements for more efficient re-sampling trials are described. In the end, the detected robust interactions are presented for three real-world GWAS studies. This chapter is partially derived from the following publication of related contents[a].

[a]Qiao Wang, Fan Shi, Andrew Kowalczyk, Richard M Campbell, Benjamin Goudey, David Rawlinson, Aaron Harwood, Herman Ferra and Adam Kowalczyk, "GWISFI: a Universal GPU Interface for Exhaustive Search of Pairwise Interactions in Case-Control GWAS in Minutes", *IEEE International Conference on Bioinformatics and Biomedicine*, Belfast, UK, 2014.

3.1 Overview

Epistatic interactions between genes are believed to be a critical component in the genetic architecture of complex diseases. Genome-wide association studies (GWAS) may be able to detect such genetic interactions indirectly, via the identification of associated SNP markers. Major obstacles to progress in this area are: the unknown nature of epistatic interactions, little understanding of the capabilities of different filtering methods, and the computational difficulties for exhaustive analysis. Various filtering methods are usually proposed in the form of pre-defined statistical tests for the purpose of measuring the association between the genotype and the phenotype. Unless a dataset is processed by an association test, it is unknown what types of interactions are favoured by such test in the first place. However, running each filter in an exhaustive manner for detection

of pairwise interactions poses significant challenges in computation since the computational complexity is quadratic to the number of SNPs. It may take weeks even months for a classical CPU-based method to run through a medium-sized GWAS data in literature, leading to an unreasonable waiting time for research. Moreover, a single exhaustive search by itself is insufficient from the point of view of stable feature selection. Multiple runs are required via the re-sampling trials to select a list of robust interactions that are effectively against the sampling variations in the training data. In this chapter, we present a GPU-based acceleration technique that makes the exhaustive search-based re-sampling procedure practical for selecting robust SNP-SNP interactions for real GWAS studies.

GWAS has been considered a fundamental instrument in unveiling the genetic etiology of non-mendelian complex diseases [47]. Along with the advancement of the genotyping technologies, a large number of diseases have been genotyped with more than 3600 GWAS studies published in recent years [96, 129]. Genotyping technologies provide a diverse range of SNP arrays for screening thousands of individuals, with the resolution ranged from nearly half a million SNPs for a typical medium-sized GWAS, such as the one from the Wellcome Trust Case Control Consortium (WTCCC) study [132], to the studies containing million-level probes (e.g. $>$ 2,000,000 SNPs) [139]. In order to reveal the underlying biological mechanisms of disease, most analytical methods use single-locus strategy to identify significantly associated SNPs, in which each SNP is examined individually for association with the disease. However interactions among loci are believed broadly contribute to complex diseases with non-negligible joint effects, while each SNP may show little effect independently [25].

Due to the large search space ($mn(n-1)/2$ for m individuals and n SNPs) of bivariate interactions, it was once considered technically infeasible to perform an exhaustive search without an access to the high performance computing facilities [38, 115, 159, 162]. As a result, researchers try to reduce the search space by performing a pre-filtering strategy as we have discussed in Chapter 2. But such methods may miss genuine pairwise interactions with very weak marginal effects [22, 25, 160], making exhaustive search an inevitable task. The spectacular progress in commodity computing technology in the last few years has led to the development of a number of algorithms capable of exhaustive

bivariate analysis not only on moderate computer clusters but also on standard desktop computers equipped with graphics processing units (GPU). In order to fully exploit the potential of these devices for medical and biotechnological research, efficient software tools are required to reduce implementation and performance difficulties allowing researchers to focus on the comparison of statistical results rather than wrestling with the intricacies of software deployment and low-level algorithm tweaking.

Table 3.1: A general runtime comparison of different bivariate interaction detection methods in Case-Control GWAS. Runtimes are purely scaled from the reported data in literature by $O(mn^2)$ for n SNPs and m samples without taking any other factors into account. The estimated results are shown for 450K and 4.5M SNPs datasets with 5K samples each. ♯ Pairwise logistic regression is used; † Corrected number to our publication. $*$ Plink 1.9 (beta 6.21) is a newer version of the original work [109, 110] available at `www.cog-genomics.org/plink/1.9/`, and the reported 3.1 hours runtime is based on the in-house run on our local desktop (CPU: Intel Core i5-6500 processor; Operating system: 64-bit Ubuntu Linux 16.04; Disk: KINGSTON SV300S3 240GB; Memory: 2× 8GB F4-2133C15-8GVR; Motherboard: Gigabyte H170-D3HP-CF; GPU: Nv Tesla K40c).

Methods	Hardware Platform	450K x 5K	4.5M x 5K
GWIS (1 filter) [37]	1 x Nv GTX 470	0.22 hour	0.92 day
SHEsisEpi [54]	2 x Nv GTX 285	21.8 hours	90.8 days
EPIBLASTER [62]	4 x Nv GTX 295	48.6 hours	202.5 days
GBOOST [155]	1 x Nv GTX 285	2.2 hours	9.17 days
GLIDE [63]	12 x Nv GTX 580	$\sim$ 5 hours	$\sim$ 20.8 days
SNPInt-GPU [147]♯	1 x Nv Tesla P100	3.57 hours	15.6 days
BiForce [42]	Cluster 128 CPU cores	0.37 hour	1.54 days
eCEO [145]	Cluster 172 CPU cores	$\sim$ 18 hours	$\sim$ 75 days
IG [19]	1 x CPU	$\sim$ 20 months†	$\sim$ 166.7 years
PLINK FastEpi 1.9 [17]*	1 x CPU (multi-threads)	3.1 hours	12.9 days
Hybrid method [148]	1 x Xilinx UltraScale KU115 FPGA + 1 x Nv Tesla P100	0.15 hour	0.625 day

In order to facilitate the epistasis research, a variety of efforts have been made to propose and efficiently implement statistical tests in the hope of detecting SNP-interactions that are significantly associated with the traits. The general strategy is to first draft a tailored test for association and then specifically implement and optimise a software for data processing, which often require an access to a dedicated computing infrastructure. This strategy ends up with different implementations with various hardware requirements. The resulting runtimes, depending on the implementation details, differed dramatically

across the methods. The reality is characterised in Table 3.1.

As can be seen from the table, the single CPU-based methods are generally impractical for processing large dataset. For the fastest PLINK 1.9 method, the processing time for a medium-sized 450K (#SNPs) data is acceptable, but its performance on 4.5M data takes 12.9 days. In comparison, the cluster machines can be used to parallelise the workload of a CPU. The best example is BiForce that consumes 1.54 days for the large dataset using 128 CPU cores. However, without a proper implementation, such as eCEO, it may take an unreasonably long time to process a 4.5M high dimensional data. For both examples, the users are required to access a cluster computing facility which is not always available to a research team. Another way of speeding up the data processing is to rely on hardware accelerators such GPU and FPGA, which are generally more expensive and require more power consumption to accommodate. But such solutions are independent from the clusters and are more cost-effective for an ordinary research team to use. However, without proper implementations and optimisations, such as SHEsisEpi and EPIBLASTER, the runtimes are still not acceptable for processing large datasets. Another consideration is their respective hardware settings. For instance, SHEsisEpi, EPIBLASTER and GLIDE require multiple GPUs, and the hybrid method [148] relies on a GPU + FPGA architecture. Other methods, such as GWIS, GBOOST and SNPInt-GPU, use single GPU for data analysis, but the runtimes varies from $\sim$ 1 day to 15.6 days. For an ordinary research group it is unable to fulfill all specified hardware requirements to use these methods.

We demonstrate in this chapter that there is no gold standard to define an association test and it is insufficient to explore a single statistical test for a given dataset since each method may produce a different set of features and there is no guarantee for those features to be regarded as valid predictors in Case-Control classification for a particular disease unless those methods are thoroughly tested. Therefore, we wish to test each method on each dataset to analyse the result, but we are unable to set up, for every method presented in Table 3.1, the exact hardware instrument locally. Moreover, we don't want to be trapped into the cycle of proposing, implementing, optimising and deploying for a novel method using a tailored hardware architecture in future because it takes excessive engineering efforts, while the results may turn out to be undesired in the end. In con-

clusion, it is expected to have a universal interface allowing easy prototyping of different methods while enabling fast data processing capacity for each method. By having such technique, we are able to minimise the research and engineering cycles; learn and refine the proposed methods; avoid the issue of incompatible input and output; increase the software compatibility and portability; minimise the varying demands on computational resources; and make different methods directly comparable.

Last but not least, a single run through the dataset is insufficient for producing stable features. From Chapter 2, we already know that the characteristics of data and the bias of the test method jointly decide the produced results. The detection of epistatic interactions makes the selection of genuine features even harder since the search space is quadratic in complexity and the generated highly significant SNP-pairs may vanish for a small variation introduced into the training data. To reduce false positives, the re-sampling techniques are widely used in bioinformatics, while for epistatic interactions detection in particular, the MDR method already adopted the cross-validation consistency as a criteria for selecting features that are robustly against the sampling variations. Obviously, the bottleneck is the expensive computations of repeated sampling trials since for each trial an exhaustive search is incurred. The runtimes shown in Table 3.1 are estimated for a single exhaustive search only. Assuming a 10-fold re-sampling is performed, then the exhaustive search is executed 10 times on 90% random samples each time, making the data processing impractical for most existing methods. Therefore, our objective in this chapter focuses on the robust interactions selection using fast GPU technique. As a result, the minimum expectation in computation is to make the exhaustive search-based re-sampling trials efficient and affordable for GWAS analyses.

To address the challenges, we summarise the desired functionalities of a qualified screening system as follows:

- Flexibility: The system should provide an interface allowing users to define their own association tests with minimum engineering efforts.
- Efficiency: The data processing platform is required to be fast enough for exhaustive search so that the detection of robust interactions is feasible via the re-sampling trials. All test methods are expected to run ultrafast via the platform.

- Independence: The system is expected to run with locally available consumer-level hardware, while the access to a large-scale computing facility (often restricted by quota and not always accessible for many researchers) should be minimised.
- Data utilisation: The data formats are expected to be unified such that multiple filtering methods use the same input and generate results in the same output format for easy comparison. The system should minimise the data transfer at any stage to reduce the overhead.

For the rest of this chapter, we first review several pioneer works related to efficient computation, highlight key concepts and then introduce our method. The proposed system successfully addressed the listed requirements to a great extent, making the robust interactions selection a practical job. In addition, we also provide extended algorithms to further improve the computational efficiency for re-sampling trials. Due to the limited time left for this research here, we leave the actual implementations into future work.

3.2 Review of Related Works and Concepts in Computation

The proposed GPU-based acceleration platform enables the fast prototyping of novel methods and the efficient data processing at the same time. To demonstrate the idea, five popular tests are implemented via the proposed platform and the processing time for each method is reduced to a very low level to enable robust interactions detection via re-samplings. The presented work leveraged from our previous GPU solution developed in [37], and we begin with an introduction to several key concepts in computation. We use the notations in Table 2.1 to formalise the discussion.

For a pair of SNPs, there are nine genotype combinations for each of the two 3×3 contingency tables, either in Case or Control, and the sample counts are distributed to the nine cells by calculating the occurrences of each genotype combination $[n_{ij}^c]_{0 \le i,j \le 2}$. Based on the filled contingency tables, many statistical tests can be built on top to derive a raw statistic to measure the association between the genotype and the phenotype. Therefore, the efficient calculation of a contingency table becomes a hurdle to the follow-up evaluation by a test method in GPU.

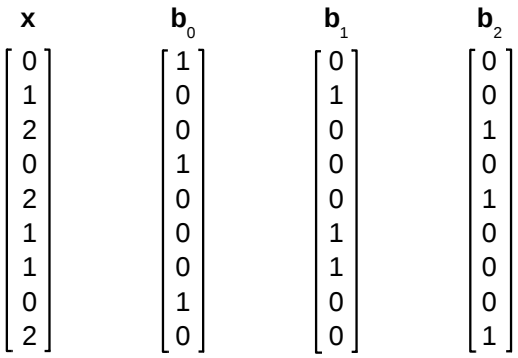

Figure 3.1: The binary data representations of a SNP.

Given a GWAS data $\mathbf{X}$ containing m samples (rows) and n SNPs (columns), a contingency table can be quickly filled with sample counts for each pair of SNPs $\mathbf{x}_a$ and $\mathbf{x}_b$ by using the binary representations of $\mathbf{X}$. Because each SNP vector $\mathbf{x}$ contains three possible genotypes $\{0,1,2\}$ across the individuals, so for each of the three genotype categories one can generate a binary vector with each bit '1'/'0' indicating the presence/absence of that genotype for a particular individual. As a result, each column vector $\mathbf{x}$ (belongs to either Case or Control group) can be converted into three binary vectors for the three genotypes, and this data representation is originally proposed by [141]. To demonstrate its idea, we illustrate three binary vectors $\mathbf{b}_0$, $\mathbf{b}_1$ and $\mathbf{b}_2$ for a SNP vector $\mathbf{x}$ in Figure 3.1. To simplify the on-going discussions, we use $\mathbf{b}$ and $\mathbf{B}$ to denote the generic representations of the possible binary vectors and the corresponding binary matrix respectively. In our previous implementation, only two binary vectors are actually needed since the counts for the third one can be simply derived by subtracting the other two from the precomputed marginal counts (stored constantly in GPU global memory). To simply the discussion, all other simplifications, optimisations and related implementation details are skipped.

Before actually calculating the cell counts, there is a necessary intermediate step to compress the binary vector $\mathbf{b}$ into a packed format $\mathbf{p}$, in which each vector element is a 64-bit unsigned integer. The conversion is simply carried out by performing the dot product between a vector $\mathbf{w} = (2^0, \cdots, 2^{63})$ and every 64 consecutive instances of the column vector $\mathbf{b}$, Figure 3.2. For simplicity, we assume that $m/64$ is divisible in all scenarios, and we can see from the figure that the binary vector $\mathbf{b}$ with m instances is converted into the vector $\mathbf{p}$ with $m/64$ integers. Correspondingly, the binary $m \times n$ matrix $\mathbf{B}$ is transformed

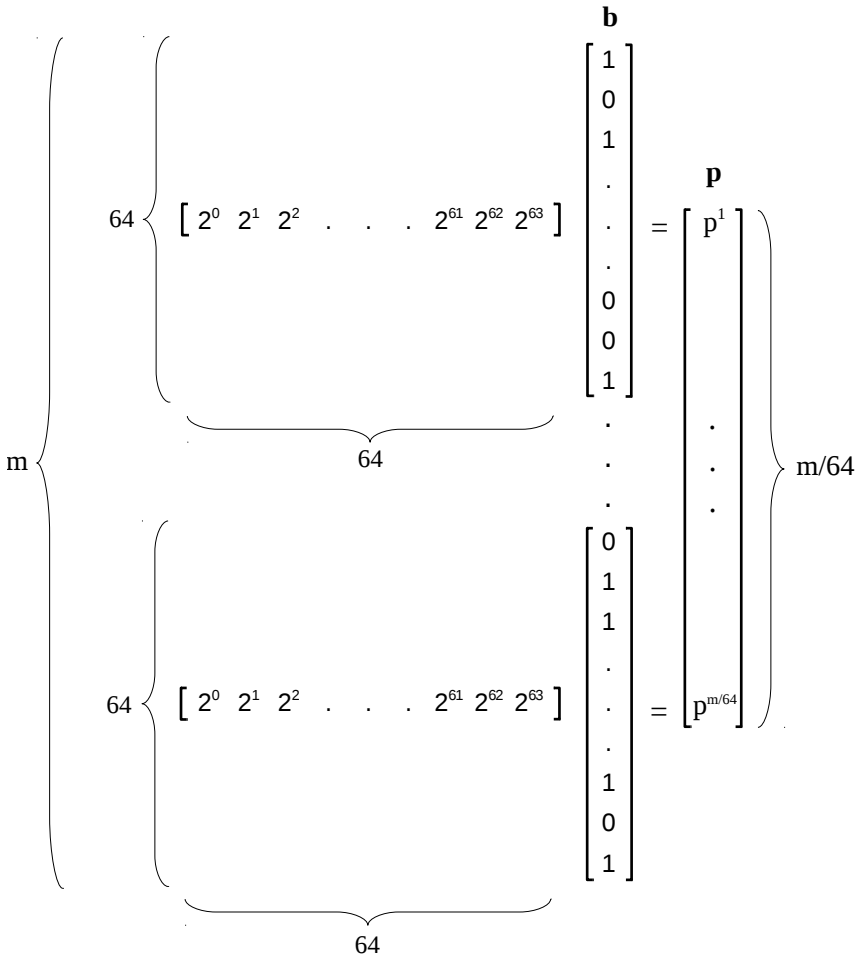

Figure 3.2: Pack a binary vector into 64-bit integer form.

into a packed $m/64 \times n$ matrix $\mathbf{P}$, Figure 3.3.

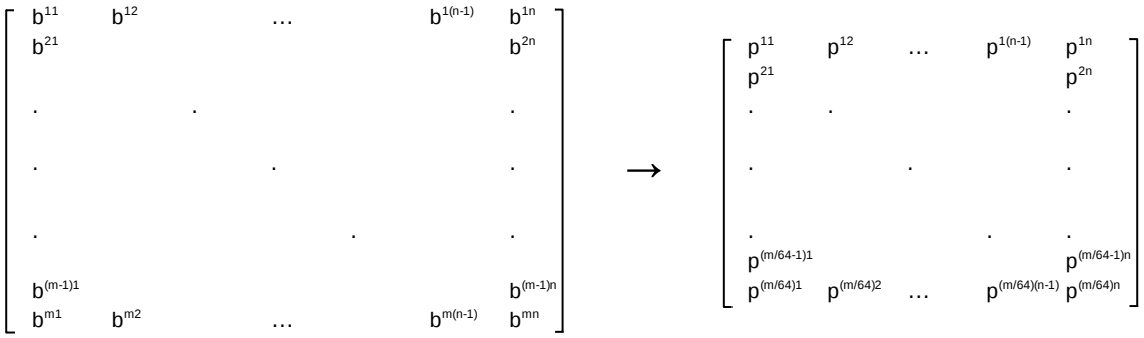

Figure 3.3: The binary m × n matrix is converted into a packed $\frac{m}{64} \times n$ matrix.

Here, each SNP vector $\mathbf{x}$ is finally represented by three packed vectors $\mathbf{p}_0$, $\mathbf{p}_1$ and $\mathbf{p}_2$

for the three genotypes. Similarly, we use $\mathbf{p}$ and $\mathbf{P}$ to denote the generic representations of three possible packed vectors and the corresponding packed matrix respectively. For instance, to evaluate the cell counts n_{ij}^c for the genotype combination $i = 0$ and $j = 1$ of a SNP-pair $\mathbf{x}_a$ and $\mathbf{x}_b$, we can refer to the vectors $\mathbf{p}_0^a$ and $\mathbf{p}_1^b$ to perform the bitwise operation AND for each pair of row elements in the two vectors and apply the 64-bit hardware population count function *__popcll* in GPU to calculate the number of ones in the integer. Finally, we sum over all $m/64$ occurrence numbers to obtain the sample frequency n_{ij}^c of that genotype combination. This example is illustrated in Figure 3.4.

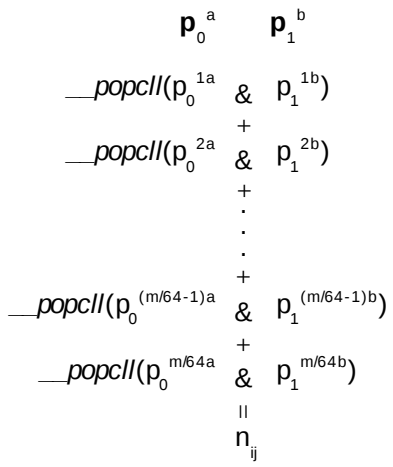

Figure 3.4: An example of calculating the sample counts n_{ij} for the genotype combination $i = 0$ and $j = 1$ of a SNP-pair $\mathbf{x}_a$ and $\mathbf{x}_b$ in GPU.

We will see shortly in the experiment section that the bitwise operations and population count instructions in building the contingency table in GPU consume most of the processing time, and we show later in the robust interactions detection section that redundant operations in this bit can be identified to save some expensive calculations to speed-up the re-sampling trials. In the following section, we introduce the proposed GPU platform that utilises this data structure to perform the calculations.

3.3 A Universal GPU Interface

The proposed GPU-based acceleration platform is called GWISFI, denoting Functional Interface for Genome Wide Interaction Search. The system is highly optimised using the

NVIDIA compute unified device architecture (CUDA) for parallel computing. GWISFI delivers a functional interface allowing users to define their customised statistical tests for the exhaustive evaluation of all SNP-pairs in Case-Control GWAS data. Any test that can be expressed in terms of contingency tables can be evaluated efficiently by GWISFI.

The interface provides a number of arguments, including the contingency table and marginal counts, as the fundamental programming elements for a given SNP-pair under evaluation. Optionally, it takes other data files as additional arguments to the interface in support of the evaluation. The interface is given in the form of a CUDA kernel function, and a user defined statistical test can be built on top of the function-provided arguments and other features supported by CUDA in standard C syntax such as flow control, looping statements, logical and bitwise operators, basic data types and arithmetic instructions and so on. Apart from that, the building of contingency tables for each SNP-pair is automatically handled by another kernel function in a very efficient way.

The key features of GWISFI are summarised as follows:

- A CUDA kernel function goes through the designated data elements in packed format and rapidly generates two 3×3 contingency tables for Cases and Controls respectively using the fast bitwise operations and hardware population count instructions. The filled tables are then used by test methods for evaluation.
- A GPU interface containing instructions for user-defined statistical tests in the form of a separate CUDA kernel function to compute the raw statistic based on the sample counts in the derived contingency tables. The calculated statistic measures the association between the genotype and phenotype and is used as the sorting key for each SNP-pair under evaluation.
- A score ranking mechanism, which is based on the CUDA Thrust library, efficiently maintains a global list of top-ranked SNP-pairs according to the statistical test being evaluated. A maximum of one million top interactions are recorded and updated in the sorting buffer throughout the exhaustive search.
- Multiple statistical tests can be executed in a sequential order after the contingency tables are created. Once the time-consuming table calculations are finished, they are stored into GPU memory and are subsequently used by different test meth-

ods to derive their respective statistics. All operations of a given test method are handled by an independent channel in GWISFI.

In the beginning of the exhaustive search, the entire GWAS dataset is binarised, packed and loaded into GPU memory because the costs of I/O and data transfer between main memory and GPU memory are costly. For a medium-sized dataset (e.g. 450K SNPs $\times$ 5K samples), a low-end laptop GPU with 1GB memory is sufficient to store the entire dataset. For a larger dataset (e.g. with 4M SNPs $\times$ 5K samples), a better GPU such as the Nv GTX Titan with 6GB memory will suffice. We tested the actual memory consumption for storing such large dataset using our Nv Tesla K40c GPU and total 5.2GB GPU memory is used for all required storage.

Given the Single-Instruction Multiple-Data (SIMD) architecture of GPU devices, efficient data processing is achieved by the simultaneous analysis of large chunks (blocks) of contiguous SNP-pairs. Instead of iterating over all possible $n \times (n-1)/2$ SNP-pairs, GWISFI traverses through all possible pairs of SNP chunks, where each chunk contains $C = n/block$ SNPs. For all SNP-pairs in each chunk-pair, contingency tables are generated by a large number of CUDA threads using the kernel function. This parallelism is essential for high-throughput computing on GPU devices. The generated contingency tables for all SNP-pairs in the current block-pair are stored into GPU global memory. A follow-up CUDA kernel function executes the user-defined statistic to score each SNP-pair utilising contingency tables calculated previously.

User-defined functions are called via CUDA kernel function pointers and take the contingency table of genotype frequencies and other optional data files as arguments. To add new statistical tests, the user writes and compiles one or more functions that conform to the functional interface specification. All code is then linked into the final executable using the NVIDIA CUDA toolkit. Ranked scores are stored as a tuple consisting of the real-valued raw statistic itself and two integer indices representing the relevant SNPs. Due to the limited storage space of GPU memory and the cost of sorting a large number of SNP-pairs, the output buffer stores a maximum of one million sorted pairs using the CUDA Thrust library. Unlike most existing methods which require a predefined hard threshold for filtering SNP-pairs in the first place, GWISFI always keeps a maximum of

one million top pairs in the buffer based on the ranking of statistic. This feature allows a manageable set of SNP pairs to be generated, and provides extra benefit in many follow up analyses and applications.

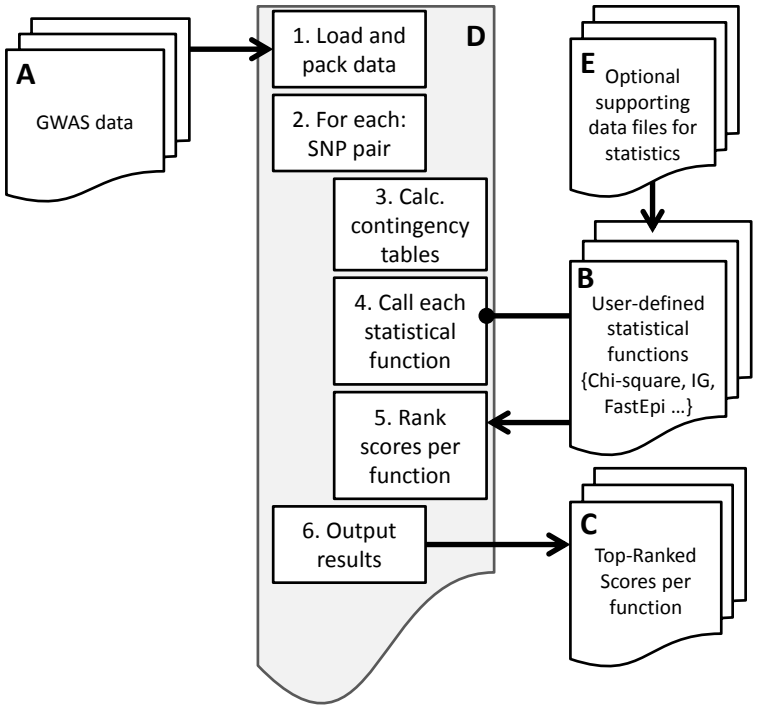

Figure 3.5: The GWISFI architecture. The system accepts GWAS data as input (A). To add a novel statistical test, the user defines one or more CUDA kernel functions (B) that accept contingency tables and other arguments describing each SNP-pair being analysed. These functions are then linked into the final executable program. The program outputs a list of top-ranked SNP-pairs for each statistical function (C). Most of the difficulty and functionality required for function evaluation is captured by the GWISFI system (D). Within the system, the data is first converted into the internal bit-mask representation and then each SNP-pair is evaluated. User-defined statistics are then evaluated for each SNP-pair, given the contingency table and other variables calculated efficiently by GWISFI. Optionally, the user can provide additional data files for use by the statistical functions (E).

Lastly, instead of computing two contingency tables (for Case and Control respectively) for each association test separately, GWISFI calculates two tables once only, and the derived genotype frequencies can be used subsequently by multiple test methods, avoiding the expensive re-calculation of tables each time. In such a case, GWISFI allocates a buffer separately to each test, maintaining a list of top-ranked SNP-pairs. The overview of the system architecture is depicted in Figure 3.5.

3.4 Runtime Improvement

To test the performance of GWISFI, we select five popular association tests from literature and implement them via the functional interface. A NVIDIA Tesla K40c GPU is used in the experiment to process a simulation dataset containing half a million SNPs and 50K samples. The raw data (matrix in text file format containing genotypes $\in \{0,1,2\}$) is 48G large in size and stored locally for analysis.

Table 3.2: Runtime improvement for several popular methods implemented via GWISFI. All numbers of the original implementations of methods are scaled from the estimated runtimes in Table 3.1. Refer to the publication and code for more implementation details (e.g. dealing with missing values, possible empty cells, etc). The '1.1' hours runtime via GWISFI is the wall-clock time, taking all computations, I/O operations, data packing, etc., into account on our desktop. The dataset with half a million SNPs and 50000 samples is used for benchmarking the performance. GWISFI runs all experiments with a single Nv Tesla K40c GPU. † Throughout the thesis, the stage one filtering is used by default for EPIBLASTER in all experiments; $*$ PLINK FastEpi 1.9 (beta 6.21).

Methods	Original Platform	Estimation 0.5M x 50K	via GWISFI 0.5M x 50K	1 x Nv Tesla K40c Speedup
GWIS (χ^2 test)	1 x Nv GTX 470	2.7 (hours)	1.1 (hours)	$\mathbf{2.45}\times$
SHEsisEpi	2 x Nv GTX 285	11.2 (days)	1.1 (hours)	$\mathbf{244.36}\times$
EPIBLASTER†	4 x Nv GTX 295	25 (days)	1.1 (hours)	$\mathbf{545.45}\times$
FastEpistasis*	1 x CPU	38.3 (hours)	1.1 (hours)	$\mathbf{34.82}\times$
IG	1 x CPU	20.6 (years)	1.1 (hours)	$\mathbf{1.64{*}10^5}\times$

The runtime of each method implemented via our platform is compared to its original implementation, with the improvement results shown in Table 3.2. As can be seen from the table, most methods received a significant speedup via GWISFI, and the processing of large dataset becomes feasible. One thing to clarify here is that the $2.45\times$ speedup estimated for the χ^2 test in our previous implementation is not a real speedup since a more advanced GPU is used here. Nevertheless, all runtimes are cut down to nearly 1.1 hours, which is a reasonable waiting time for research. In addition, GWISFI doesn't rely on complicated hardware settings. Instead, it requires a single GPU only, which is affordable to an ordinary research team. A further speedup can be expected once more advanced GPUs with more CUDA cores are allocated to the system.

Table 3.3: Runtime decomposition for the dataset with 0.5M SNPs and 50K samples. In addition to the result of each individual filter, we also present the runtime using all 5 filters in a single run via GWISFI. In this mode, the contingency table is derived once only and then used subsequently by 5 filters. $*$ PLINK FastEpi; ‡ All other overheads including the data loading, packing, transferring to GPU, initialising the storage and buffer for each individual filter, etc., other than the actual computation.

Filtering Methods	Computing Time (in minutes)	0.5M x 50K data Loading Time‡ (in minutes)	Total (in hours)
GWIS (χ^2 test)	52.21	14.49	~ 1.1
SHEsisEpi	52.75	14.74	~ 1.1
EPIBLASTER	52.52	14.55	~ 1.1
FastEpistasis*	52.29	14.51	~ 1.1
IG	52.79	14.47	~ 1.1
All five together	62.96	15.44	~ 1.3

Next, we decompose the total runtime into loading time and computing time, trying to identify the main bottleneck of the computation. The time decomposition for the large synthetic data is shown in Table 3.3. We can see from the table that the computing time generally takes about $52.5/67 \simeq 78\%$ of the total time if every method is executed independently on our desktop. In such a case, all methods go through the entire procedure of data processing. As a result, the large input data is loaded five times with a total of ~ 73 minutes spending on the data loading and packing, which is a non-negligible overhead. Assuming hundreds of re-sampling trials are performed to select robust interactions, then this overhead becomes a significant obstacle. In summary, the data loading, packing and transferring between disk, main memory and GPU memory should be minimised, and this is one of the major practical considerations of proposed robust interaction selection algorithms via re-sampling trials.

Meanwhile, if all five methods are executed together in one go via GWISFI, then the entire dataset is loaded once only. On average, each method consumes 14.55 minutes of the loading time but such overhead for all five methods together is 15.44 minutes, which is about 1 minute longer. The reason of this ~ 1 minute discrepancy is due to more time

are spent on extra overheads, such as preparing five buffers instead of a single one. The related costs slightly increased the total loading time.

On the other hand, if we take a closer look at the computing time for all five methods together, then the total 62.96 minutes runtime is comprised of 1 unit of contingency table calculations and 5 units of statistical tests. In comparison, the average computing time of each method independently is 52.5 minutes which is made up of 1 unit of contingency table calculations and 1 unit of a statistical test. Consequently, if we subtract 52.5 from 62.96, then we have $62.96 - 52.5 = 10.46$ minutes for total 4 statistical tests, and each test on average consumes $10.46/4 = 2.6$ minutes. If we further subtract 2.6 from 52.5, then the time spent on building the contingency tables is $52.5 - 2.6 = 49.9$ minutes on average, which means $49.9/52.5 = 95\%$ of runtime is spent on the contingency table calculations by a single method in GPU for this dataset.

In conclusion, the contingency table building, which relies on the bitwise operations and hardware population count instructions, becomes a major bottleneck in GPU computing. Although it is hard to further decrease the overhead of this type of computation in a single run, we show in the following section that redundant operations can be identified in the re-sampling procedure for robust interactions selection.

3.5 Select Robust Interactions via Re-samplings for Large Data

Recall from previous discussions that a single run through the data may lead to false positives, and we wish to select only those SNP-pairs that are robustly against the variations introduced into the training data. A straightforward method is to adopt a re-sampling procedure, in which a portion of data are randomly selected as the training set for each trial and an exhaustive search is performed on that training set to report a list of top-ranked SNP-pairs. By repeating the re-sampling procedure multiple times, we select those SNP-pairs that consistently appeared in multiple folds of re-sampling trials as the robust ones. However, if the experiment is performed directly using the basic version of GWISFI, then the data loading, packing and transferring to GPU memory is repeated for each re-sampling trial. As we have shown in the previous section that it becomes a

noticeable problem for large datasets. In this section, the term *large data* refers to a dataset with a large volume of samples.

Below, we provide two extension algorithms of GWISFI to improve the efficiency of selecting robust SNP-SNP interactions via re-sampling trials. We try to address previously mentioned two bottlenecks, namely, the contingency table calculations and the loading and packing for large datasets.

3.5.1 Avoid Repeated Loading of Large Data

The first challenge is to avoid repeated loading of a large dataset when the re-sampling trials are performed hundreds even thousands of times. Instead of going through the complete cycle of loading, binarising and packing of the dataset at each trial, a desired characteristic is to follow these steps once only and constantly keep the packed data in GPU memory, where a light-weight operation can be performed directly on the packed data to identify a portion of randomly selected samples in the packed format as the training set for that trial. After that, an exhaustive search is applied on that training set to produce a list of top-ranked SNP-pairs for that trial.

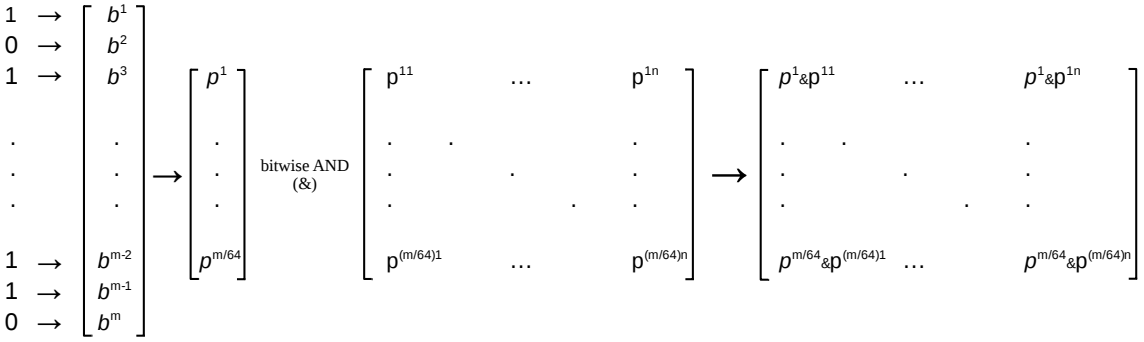

Figure 3.6: Randomly disable a portion of rows in a packed GWAS dataset to generate a training set $\mathbb{P}$.

Assuming we wish to use 75% random samples for the exhaustive search at each re-sampling trial, one simple way is to disable the remaining 25% from the packed data by replacing certain rows encoded in the packed data with zeros. To achieve that, we first determine that ratio $\alpha = 0.75$ as the proportion of samples to be used as the training set. After that, we create a list of randomly generated ones and zeros, ensuring the ones take

that proportion α in the list. Next, this binary indicator vector $\boldsymbol{b}$ is packed into $\boldsymbol{p}$, and we perform the column-wise bitwise operation AND (&) between $\boldsymbol{p}$ and previously packed original data $\mathbf{P}$ to generate the desired training set $\mathbb{P}$, Figure 3.6.

To better elucidate the principle, consider two 64-bit words p_0^a and p_1^b representing the genotype combination '0-1' for a pair of SNPs '$\mathbf{x}_a$' and '$\mathbf{x}_b$'. The frequency of this genotype combination can be simply obtained by performing $\_\_popcll(p_0^a \,\&\, p_1^b)$ in GPU. Given a 64-bit indicator word p, we can generate two other words $p_0^a \,\&\, p$ and $p_1^b \,\&\, p$. Certain rows in two new words are marked as zeros because of p and consequently they are deducted from the total frequency count. By performing $\_\_popcll((p_0^a \,\&\, p) \,\&\, (p_1^b \,\&\, p))$, only remaining rows (training set) are actually in effect in counting the occurrences of that genotype combination.

Therefore, to utilise a portion of random samples as the training set for a trial of exhaustive search, we only need to create a randomly generated indicator vector (with ones and zeros taking proportions of α and $1-\alpha$ respectively) in the packed format and perform the bitwise operation AND with the preserved data $\mathbf{P}$, such that a training set is derived for that trial because $1-\alpha$ of total rows are replaced with zeros and are excluded from the counting. If the re-sampling trial is executed many times, we never need to load a large number of samples from the beginning to create the training set. The following Algorithm 1 is depicted to avoid the frequent data loading in re-samplings.

3.5.2 Redundant Calculations in Contingency Table Building

Another improvement is to identify redundant calculations when building the contingency tables in re-samplings. Consider a 10-fold re-sampling scenario for the time being, the basic idea is to first calculate the contingency table T using all samples, and then calculate the contingency table $T_k^{'}$ for each fold k separately. After that, we can calculate the statistic $S_k(\Delta T_k)$ based on the sample counts in the discrepancy table ΔT_k, where $\Delta T_k = T - T_k^{'}$, which means that the genotype frequencies in $T_k^{'}$ of fold k are deducted from the total counts in T. By averaging over all $K = 10$ folds $\overline{S} = (\sum_{k=1}^{K} S_k(\Delta T_k))/K$, we obtain the sorting key $\overline{S}$ for the SNP-pair under evaluation.

To implement the idea, we can shuffle the sample order of the original GWAS data to

Algorithm 1 Robust interaction selection by re-samplings via GWISFI, with frequent data loading and packing avoided.

Require: GWAS data $\mathbf{X}$ with m rows (samples) and n columns (SNPs).

1: Determine a set of filters F for evaluation.
2: Set a ratio $\alpha \in (0,1)$ for the training samples out of the total samples.
3: Set chunk size $block$, resulting in $C = n/block$ chunks, numbered $\in \{1, \cdots, C\}$.
4: Set total re-sampling trials $\mathcal{R}$.
5: Generate the binary data $\mathbf{B}$ from $\mathbf{X}$ and pack into 64-bit integer data $\mathbf{P}$. (Figure 3.3)
6: Store $\mathbf{P}$ into GPU memory.
7: **for** $r \in \{1, \cdots, \mathcal{R}\}$ **do**
8: Generate a vector $\boldsymbol{b}$ containing $\alpha \times m$ '1' and $(1-\alpha) \times m$ '0' in random order.
9: Pack the binary vector $\boldsymbol{b}$ into a 64-bit integer vector $\boldsymbol{p}$; store $\boldsymbol{p}$ into GPU memory; and perform $\boldsymbol{p}$ & $\mathbf{P} = \mathbb{P}$. (Figure 3.6)
10: Initialise the dynamic sorting threshold θ_ψ for each filter $\psi \in F$.
11: **for** $A \in \{1, \cdots, C\}$ **do**
12: **for** $B \in \{A, \cdots, C\}$ **do**
13: **for** each pair of SNPs $\mathbf{x}_a$ in chunk$_A$ and $\mathbf{x}_b$ in chunk$_B$ ($\forall a < b$) in parallel **do**
14: Generate the contingency table $T(\mathbf{x}_a, \mathbf{x}_b)$ from $\mathbb{P}$ and store into GPU memory by a CUDA kernel function of GWISFI.
15: **for** each filter $\psi \in F$ **do**
16: Read $T(\mathbf{x}_a, \mathbf{x}_b)$; execute ψ based on the counts in $T(\mathbf{x}_a, \mathbf{x}_b)$; and store SNP_ID($\mathbf{x}_\mathrm{a}$), SNP_ID($\mathbf{x}_\mathrm{b}$) and the calculated statistic $S(T(\mathbf{x}_a, \mathbf{x}_b))$ that exceeds the θ_ψ into the buffer $s(\psi)$ by a CUDA kernel function of GWISFI.
17: **end for**
18: **end for**
19: Maintain a global list of top-ranked SNP-pairs in $s(\psi)$ for each ψ and update each θ_ψ using the CUDA Thrust library, adopted by the GWISFI platform.
20: **end for**
21: **end for**
22: **for** each filter $\psi \in F$ **do**
23: Output the list of top-ranked SNP-pairs l_ψ^r from $s(\psi)$ for the current trial r.
24: **end for**
25: **end for**

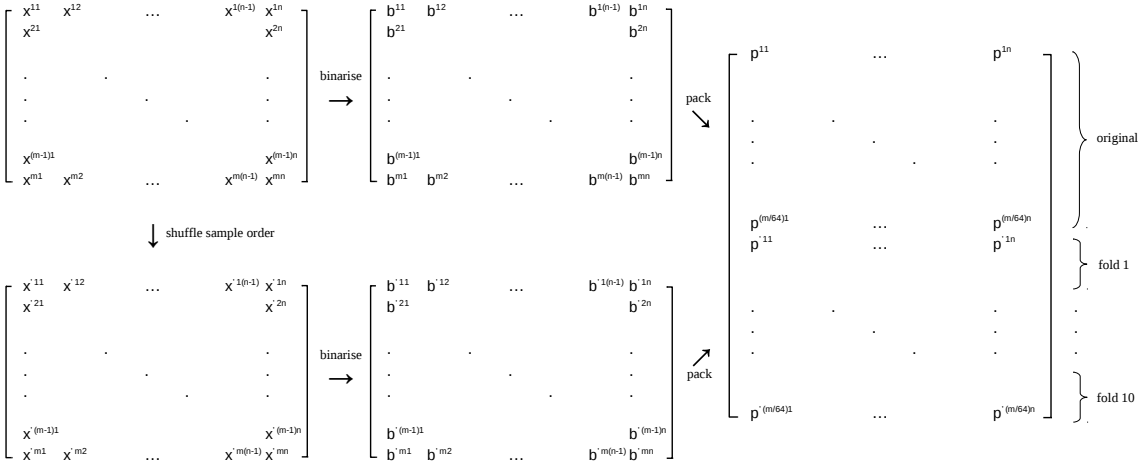

Figure 3.7: Append the shuffled samples to the original data to remove redundant calculations in contingency table building for re-sampling trials. Ten-folds of shuffled samples are used as an example, assuming sample size is large enough and the generated $\mathbf{P}^{'}$ is divisible into $K = 10$ folds.

derive a shuffled version $\mathbf{X}^{'}$, binarise it as $\mathbf{B}^{'}$ and pack into $\mathbf{P}^{'}$ (assuming sample size is very large and $\mathbf{P}^{'}$ is divisible into $K = 10$ folds). Both $\mathbf{P}^{'}$ and $\mathbf{P}$ are stored in GPU memory, and previously described idea is applied to the packed data.

Assuming 1 unit of computing time is required to build the contingency table using all samples. Correspondingly, 0.9 unit of time is consumed to build the contingency table using 9-folds of the whole data since the runtime decreases linearly as the number of packed words, then total 10×0.9 unit $= 9$ units of time are spent on contingency table building for the re-sampling procedure executed 10 times independently. By using proposed procedure, in comparison, 1 unit of time is consumed to build the original contingency table T using all samples, while 10×0.1 unit $= 1$ unit of time is required to create 10 tables $T_k^{'}$ for the 10 folds respectively. In total, only ~ 2 units of time are used to generate all 10 discrepancy tables ($\Delta T_k = T - T_k^{'}$ for $k \in \{1, \cdots, 10\}$). Thus, the efficiency of building tables for 10-fold re-samplings is improved.

This simplification of contingency table calculations is depicted in the Algorithm 2. One thing to note here is that the two extension algorithms differ in ways of ranking. The SNP-pairs in the first algorithm are sorted based on their original statistics at each independent trial of re-samplings, while the second algorithm sorts each SNP-pair based on a score averaged over K (calculated from the discrepancy tables).

Algorithm 2 Robust interaction selection by re-samplings via GWISFI, with redundant calculations reduced in contingency table building.

Require: GWAS data $\mathbf{X}$ with m rows (samples) and n columns (SNPs).

1: Determine a set of filters F for evaluation.
2: Set the number of folds K.
3: Set chunk size $block$, resulting in $C = n/block$ chunks, numbered $\in \{1, \cdots, C\}$.
4: Generate the binary data $\mathbf{B}$ from $\mathbf{X}$ and pack into 64-bit integer data $\mathbf{P}$. Randomly shuffle the rows of $\mathbf{X}$ into $\mathbf{X}^{'}$; generate $\mathbf{B}^{'}$; pack into $\mathbf{P}^{'}$; and divide $\mathbf{P}^{'}$ into K folds $\{\mathbf{P}^{'}(1), \cdots, \mathbf{P}^{'}(K)\}$. (Figure 3.7)
5: Store $\mathbf{P}$ and K folds of $\mathbf{P}^{'}$ into GPU memory.
6: Initialise the dynamic sorting threshold θ_ψ for each filter $\psi \in F$.
7: **for** $A \in \{1, \cdots, C\}$ **do**
8: **for** $B \in \{A, \cdots, C\}$ **do**
9: **for** each pair of SNPs $\mathbf{x}_a$ in chunk$_A$ and $\mathbf{x}_b$ in chunk$_B$ ($\forall a < b$) in parallel **do**
10: *// The following instructions are performed by the CUDA kernel functions of GWISFI*:
11: Generate contingency tables $T(\mathbf{x}_a, \mathbf{x}_b)$, $T_1^{'}(\mathbf{x}_a, \mathbf{x}_b), \cdots, T_K^{'}(\mathbf{x}_a, \mathbf{x}_b)$ from $\mathbf{P}$, $\mathbf{P}^{'}(1)$, $\cdots$, $\mathbf{P}^{'}(K)$ respectively and store into GPU memory.
12: **for** each filter $\psi \in F$ **do**
13: Read $T(\mathbf{x}_a, \mathbf{x}_b)$.
14: **for** $k \in \{1, \cdots, K\}$ **do**
15: Read $T_k^{'}(\mathbf{x}_a, \mathbf{x}_b)$; calculate $\Delta T_k = T(\mathbf{x}_a, \mathbf{x}_b) - T_k^{'}(\mathbf{x}_a, \mathbf{x}_b)$; execute ψ based on the counts in ΔT_k; and derive the statistic $S_k(\Delta T_k)$.
16: **end for**
17: Calculate the average $\overline{S} = (\sum_{k=1}^{K} S_k(\Delta T_k))/K$ and store SNP_ID($\mathbf{x}_\mathrm{a}$), SNP_ID($\mathbf{x}_\mathrm{b}$) and $\overline{S}$ that exceeds the θ_ψ into the buffer $s(\psi)$.
18: **end for**
19: **end for**
20: Maintain a global list of top-ranked SNP-pairs in $s(\psi)$ for each ψ and update each θ_ψ using the CUDA Thrust library, adopted by the GWISFI platform.
21: **end for**
22: **end for**
23: **for** each filter $\psi \in F$ **do**
24: Output the list of top-ranked SNP-pairs l_ψ from $s(\psi)$.
25: **end for**

3.6 Learning Robust Interactions for Real GWAS data

In this section, we apply the exhaustive search-based re-samplings on real GWAS studies to present top robust interactions and their re-sampling consistency. In the experiment, three diseases, namely Type 1 diabetes (T1D), Rheumatoid arthritis (RA) and Crohn's disease (CD), are selected from the Wellcome Trust Case Control Consortium[1] (WTCCC) [132] to produce results.

Firstly, we perform the quality control for each GWAS study using PLINK software with setting "–geno 0.1 –mind 0.1 –maf 0.05 –hwe 0.001", which ends up with $\sim$ 350K SNPs for each dataset. For each of them, total 300 re-sampling trials are performed. For each trial, 75% of samples are randomly selected from the data as the training set, and all five filters χ^2, IG, FastEpi, SHEsisEpi and EPIBLASTER are applied on the training set via GWISFI to select top SNP-pairs. As a result, each filtering method generates 300 lists of top SNP-pairs for each study. In total, there are 300 trials $\times$ 3 studies $\times$ 5 filters $=$ 4500 exhaustive search performed locally using our Tesla K40c GPU.

For each 'study-filter' combination, we select 10000 top-ranked SNP-pairs from each trial; identify the unique pairs from 300 $\times$ 10000 candidates; and assign a re-sampling consistency to each unique pair by calculating the ratio of its frequency out of total 300. Finally, all unique pairs are sorted from the largest to the smallest using the re-sampling consistency as the sorting key. From the sorted pairs, we visualise top 50 robust interactions in Figure 3.8 using the RCircos package [157]. To generate plots by RCircos, we first convert the WTCCC SNPs from genome coordinate system hg17 to hg19 using the on-line version of *liftOver* tool `https://genome.ucsc.edu/cgi-bin/hgLiftOver` because RCircos takes the GRCh37/hg19 as the default genome coordinate, whereas the WTCCC data use NCBI135/hg17 to index SNPs.

From Figure 3.8, it is not surprised to see that different filters generate dramatically different robust interactions across the genome since the bias of each filter is obvious. In such a case, we would like to know, from all reported robust interactions, which SNP-interactions generated by which filter are valid predictors in Case-Control classification.

[1] A full list of the investigators who contributed to the generation of the data is available from `www.wtccc.org.uk`. Funding for the project was provided by the Wellcome Trust under award 076113 and 085475.

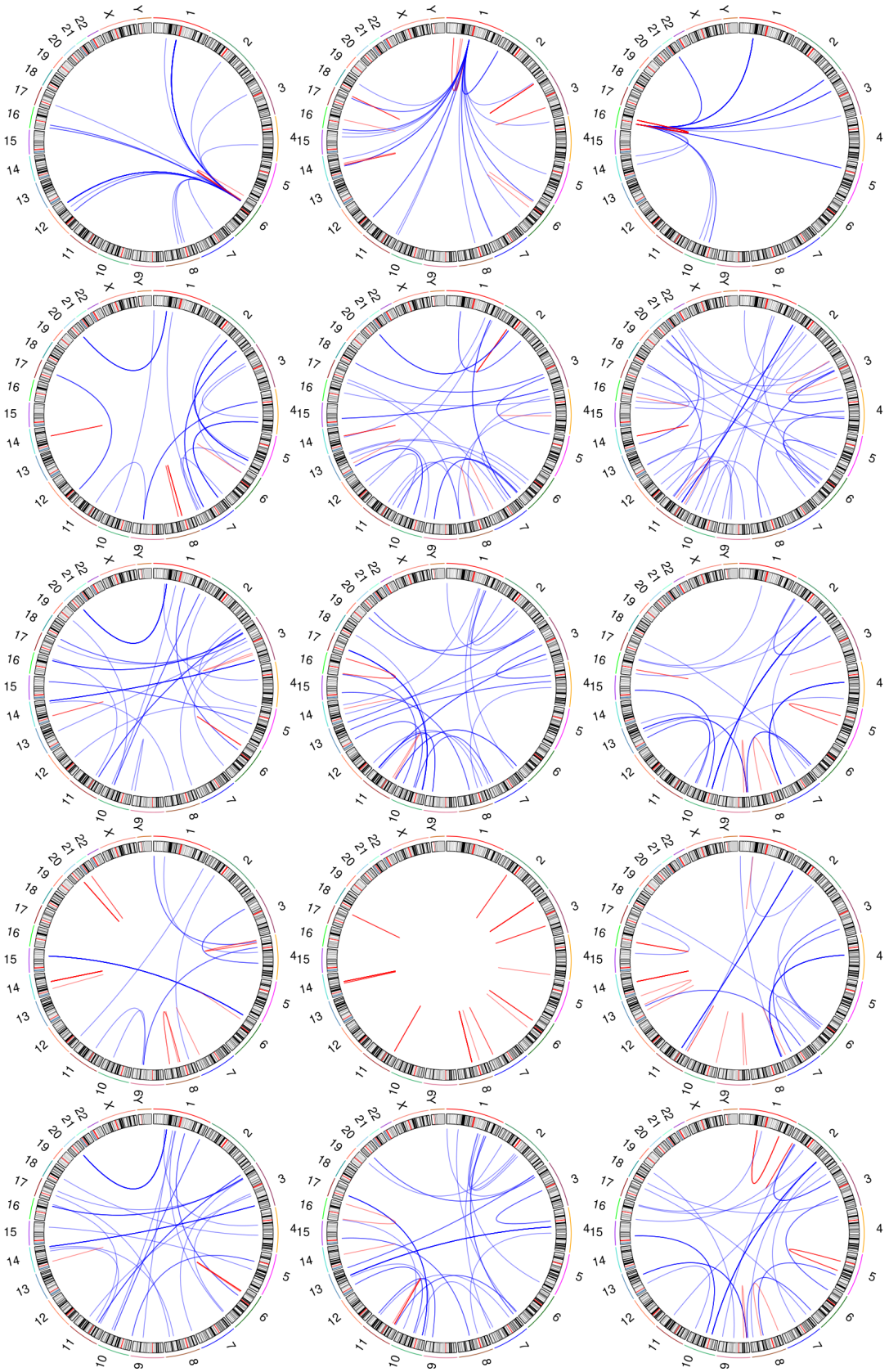

Figure 3.8: Top 50 robust interactions of 5 different filters for 3 GWAS studies. From left to right: T1D, RA, CD. From top to bottom: χ^2, IG, FastEpi, SHEsisEpi and EPIBLASTER.

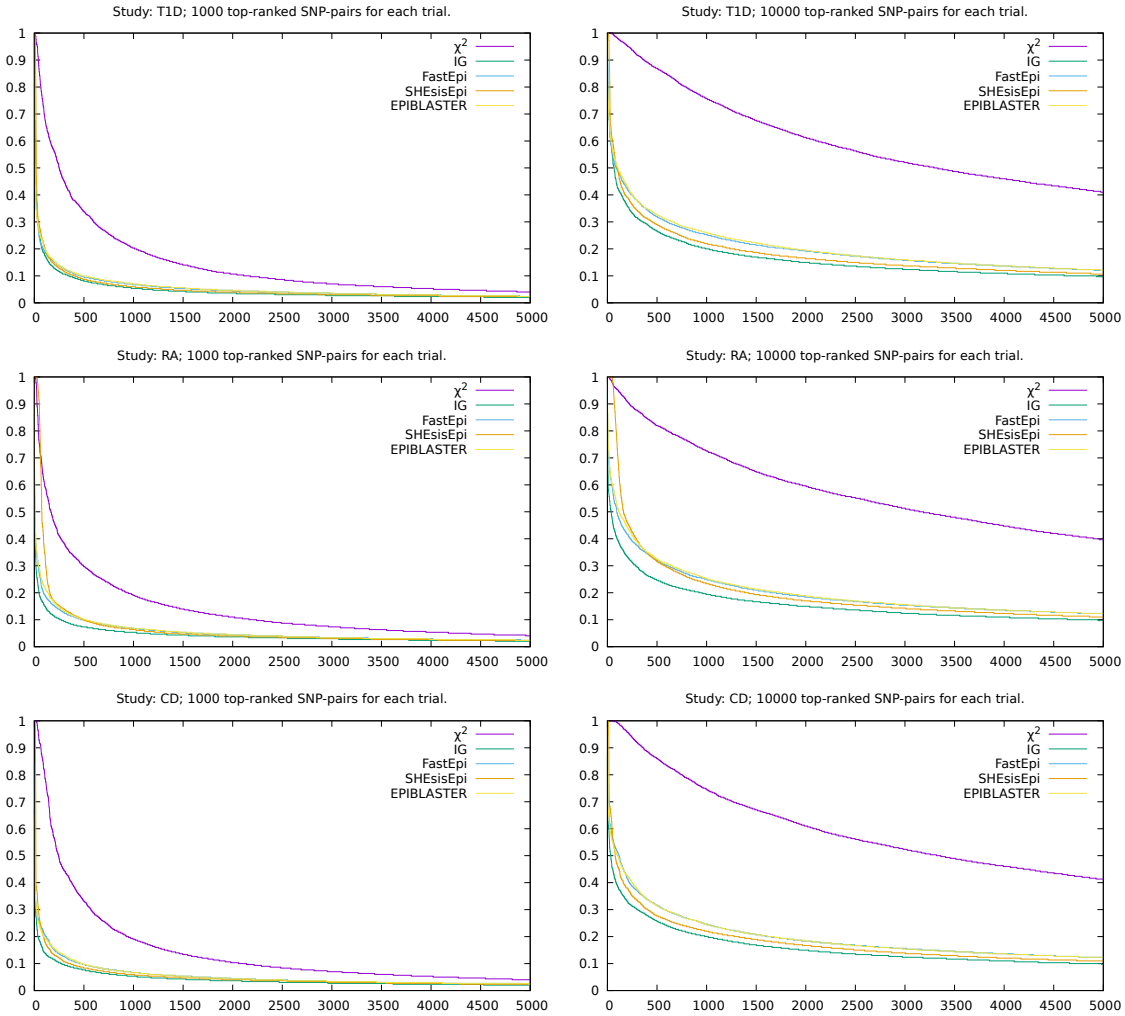

Figure 3.9: Re-sampling consistency of top 5000 robust SNP-pairs for 3 GWAS studies. For each re-sampling trial, 1000 or 10000 top-ranked SNP-pairs are used to derive the final list of robust interactions.

We response to this question by showing the prediction evaluation results in Chapter 5. At this stage, we conclude that the types of robust interactions picked by each filter are unknown to us *a priori* until all of them are thoroughly tested on real GWAS studies. Thus, a fast screening tool such as GWISFI is necessary for running those methods.

Next, we investigate the re-sampling consistency for the top robust pairs to give a general idea of how well these interactions are survived from the sampling variations. Recall from the previous analysis, we selected 10000 top-ranked SNP-pairs from each trial and calculated the re-sampling consistency for each unique one to finally obtain a

list of sorted robust interactions. Here, we give an additional option of top 1000 pairs selected for each trial, focusing on more significant ones as well. The same calculations are applied with the consistency results shown in Figure 3.9. In the figure, the left and right plots correspond to the consistency results of top 5000 unique robust pairs, generated from total 300 trials using either 1000 or 10000 top-pairs for each trial. From those plots, we see that the re-sampling consistency quickly deteriorated as the number of robust interactions increased. In other words, those interactions along the right tails of the curves are no longer robust enough and only a small portion of SNP-interactions are survived from the sampling variations. Although the consistency results using 10000 top-pairs at each trial generally perform better, the trends of two options are similar. In conclusion, the re-sampling-based robust interactions selection is an inevitable step to detect stable features, and a proper determination of top robust interactions is the key to avoid unstable features being included into the candidate feature set for further analyses.

3.7 Summary

In this chapter, we have introduced the high performance computational method of selecting robust interactions via a GPU-based screening platform. Because a single exhaustive search with a particular association test is insufficient, the selection of robust interactions via re-sampling trials becomes a key step to follow-up analyses. In response, the proposed GPU platform enables the fast exhaustive search using customised test methods. Due to the high efficiency of proposed GPU method, the exhaustive search-based re-sampling trials become feasible for selecting robust interactions, and the extended algorithms can further improve the re-sampling efficiency to process large datasets.

Chapter 4

Learning Discriminative Signals by Shallow Model

IN this chapter, a shallow neural network model is proposed, enabling both the effective binary classification and high-level class-specific feature learning at the same time. Throughout the experiments, the learned features are shown to be reasonable for the classification task and the predictive signals can be simply distinguished from the noise with very little computational overhead, making it a useful tool of learning nonlinear interaction features for Case-Control discrimination in GWAS. Various simulation tests are performed to demonstrate different aspects of proposed model. This chapter is partially derived from the following publication of related contents[a].

[a]Qiao Wang, Sylvia Young, Aaron Harwood and Cheng Soon Ong, "Discriminative Concept Learning Network: Reveal High-level Differential Concepts from Shallow Architecture", *International Joint Conference on Neural Networks*, Killarney, Ireland, 2015.

4.1 Overview

Learning epistatic interactions as effective predictors for Case-Control classification requires a predictive model to be flexible enough to realise various nonlinear and linear interaction patterns. A potentially qualified model is challenged by the objective from different aspects such as nonlinear approximation ability, interpretability, stability and computational efficiency. In this chapter, we discuss related factors and propose a novel neural network model addressing all challenges to some extent. Various aspects of proposed method are discussed with detailed experiment results shown in this chapter. Through a number of simulation tests, the method is demonstrated to be capable of realising various nonlinear discriminative patterns. The method is shallow in nature but able to perform effective classification and learn high-level class-specific features at the same time. Due

to its explicit nonlinear mapping capacity and simplicity in computation, the proposed model can be utilised by a simple procedure to efficiently distinguish the signals from the noise with minimum computational cost.

Firstly, the nonlinear nature of epistatic interactions poses significant challenges for a predictive model to flexibly recognise and distinguish them from the noise. The level of interactions in real world is not restricted to binary, ternary, but possibly extended to higher order interactions. An ideal method should not assume a specific order or certain types of interactions, but required to be capable of learning various interaction patterns for effective Case-Control discrimination. For many explicitly defined statistical tests or other ad-hoc protocols, it is highly doubtful that these routines are flexible enough to adapt to all types of interactions. In fact, some methods require manually revising a previous version of statistic for the purpose of adapting to higher order interactions or dealing with the curse of dimensionality problem.

Due to the universal approximation capacity of artificial neural networks, these nonlinear models have potentials and shed light on effective learning of nonlinear interaction features without assuming any certain types of interactions in the first place. As it has been shown in Chapter 2, applying a neural-net model directly to this task is still nontrivial. For a qualified neural network method, a number of desired characteristics are highlighted as follows:

- Classification capacity: For a given set of input features and a well-determined hyperparameters setting, the model should realise the patterns from the input data and derive a trained model for which an effective prediction can be made to a set of test samples. The prediction performance is a criterion assessing whether the model successfully learns from the training data, and the model is considered predictive when it performs an effective classification on the test data. Although the research topic aims at proposing a flexible model that effectively learns interaction features for effective Case-Control discrimination, the model's classification ability, on the other hand, is an indicator of assessing whether those learned features are meaningful. In other words, if the model is unable to perform a quality classification, then it is hard to believe that the learned interactions are valid predictors.

Therefore, as a prerequisite, it is expected to see that the proposed method shows similar prediction performance to the state-of-the-art methods.

- Feature learning ability: We have seen earlier that a feature learning procedure can be coupled with a predictive model (e.g. SVM-RFE), or be part of the model-training itself (e.g. interpreting weights of a neural-net by a measuring criterion). For a desired neural net method in particular, we expect that the features are properly scored by interpreting the learned weights such that the signals and the noise are marked in a reasonable way. The derived feature score should act as a proxy, correctly reflecting a feature's importance/contribution to the binary classification. Based on those scores, the model should also allow a simple procedure to automatically distinguish the signal set from the noise set without incurring overly expensive computations. Once they are determined, the false positives and false negatives should be minimised respectively in the two sets.
- Interpretability: Neural-nets are considered black boxes for several reasons. For instance, the learning procedure is subject to the random initial conditions; the training process is dynamic and is hard to predict for its behaviour; and the gradient-based optimisation may converge to local minima. Multiple layers of learning is frequently used in neural-net training and usually end up with a hierarchy of learned weights. Because of the layer-wise complexity, it is hard to derive simple functions to characterise the information flow from the input layer to the output layer. By simplifying the hierarchy, we expect the model to be as shallow as possible to increase the possibility of deriving an easier explanation of how the classification decision is made by the learned weights. Another observation window is to visualise the scored features. Given a binary classification problem, it is expected to see that the overall feature scores show high-level discriminative information that is human understandable. The derived feature scores should clearly distinguish as much discriminative signals as they can from the noise. As a result, the visualisation of those scores triggers the concept level cognition of human for the presented binary classification problem. Once the model behaviour is shown to be agree with the human understanding, the trust to the model is enhanced and the

learned signals are more convincing to be recognised as valid predictors.

- Stability: Due to the randomness of neural-nets, the intrinsic variations of results generated by multiple runs are expected to be a challenge. Usually, a neural-net model is required to run multiple times and the results are averaged to draw a conclusion. As we have introduced in Chapter 2, some neural network method produces inconsistent results from one trial to the other, leading to an incorrect conclusion of feature rank in the end. Although some level of systematic discrepancies between different runs may be tolerated to some extent, the ranking order among different features should not change dramatically for each trial. This is a fundamental requirement of generating reliable feature scores, and is essential for a follow-up analysis such as informative features extraction. In short, a qualified neural net-based feature selection method should take the stability issue into account. With relatively stable feature scores being produced, less repeated runs are required to confidently conclude for those features.
- Computational efficiency: A computationally efficient neural net model is critically needed under a large experiment setting. Several factors such as the dimensionality of the input, the complexity of the network architecture, the number of training epochs, etc., jointly decide the computational cost for a single trial of the experiment. However, expensive cross-validations are commonly used in machine learning for fine-tuning the hyperparameters. For each trial, the model goes through a complete training procedure. Thus, a large number of cross-validation trials significantly add up to the total computational cost. In this situation, the deep learning models are infeasible for this type of experiment due to the computational barrier. In contrast, a shallow model with highly simplified architecture could be an answer to an expensive experiment.

In this chapter, we propose a shallow neural network model called *discriminative concept learning network* (DCLN) in the hope of addressing the challenges and acquiring the listed capabilities. The method is intrinsically simple, and enables an effective binary classification and a stable feature learning at the same time. Its classification capacity is shown to be competitive to the state-of-the-art method. With the support of sufficiently

good classification ability, the method's feature learning capacity, which is our main objective, can be used to extract predictive signals. We demonstrate in this chapter that the learned features are high-level, class-specific in nature and are easily understandable to human, increasing the confidence of learned features. A number of simulation tests are performed to demonstrate different aspects of proposed method.

In the following sections, we first present the theoretical details of the method, and then demonstrate its nonlinear approximation ability using four nonlinear simulation datasets. After that, we compare its classification capacity to the SVM method using seven simulated interactions studies. Although we are not exceeding the state-of-the-art in the simulation tests, the proposed method is shown to be highly competitive. More importantly, with sufficient level of classification capability, the method certainly learns from the data and hence the generated high-level discriminative features are worthwhile checking for their potential values in applications.

Next, we demonstrate with a number of computer vision exercises first, showing that the learned features are agree with the human's understanding. This implies that the model learns and identifies key features for a binary classification problem in a way similar to humans. After that, we show how the predictive signals can be extracted from the scored feature by a light-weight procedure requiring very little computational cost. Finally, we demonstrate that the same procedure can be applied to learn nonlinear interaction features from synthetic datasets. We leave real GWAS tests in Chapter 5.

4.2 Method

In this section, we first introduce the theoretical foundation of proposed method and then give a detailed algorithm.

4.2.1 Theory

In this section, we describe the proposed DCLN method in a binary classification setup such as Case-Control classification. For a given training dataset $\mathcal{D} = \{\mathcal{X}_n, y_n\}_{n=1}^{N}$ containing N individual training instances, each instance $\mathcal{X}_n \in \mathbb{R}^{\alpha}$. In stead of using this

vector directly in the proposed method, we denote a generic input vector for such instance $\mathbf{x} = (+1, x_1, \ldots, x_\alpha)^T$ as an $(\alpha+1)$-dimensional input vector: $\forall \mathbf{x}_n \in \mathbb{R}^{\alpha+1}$, where $+1$ is an appended bias unit fixed for all instances at the input layer, and y is the class label of binary category $y \in \{\mathcal{A}, \mathcal{B}\}$.

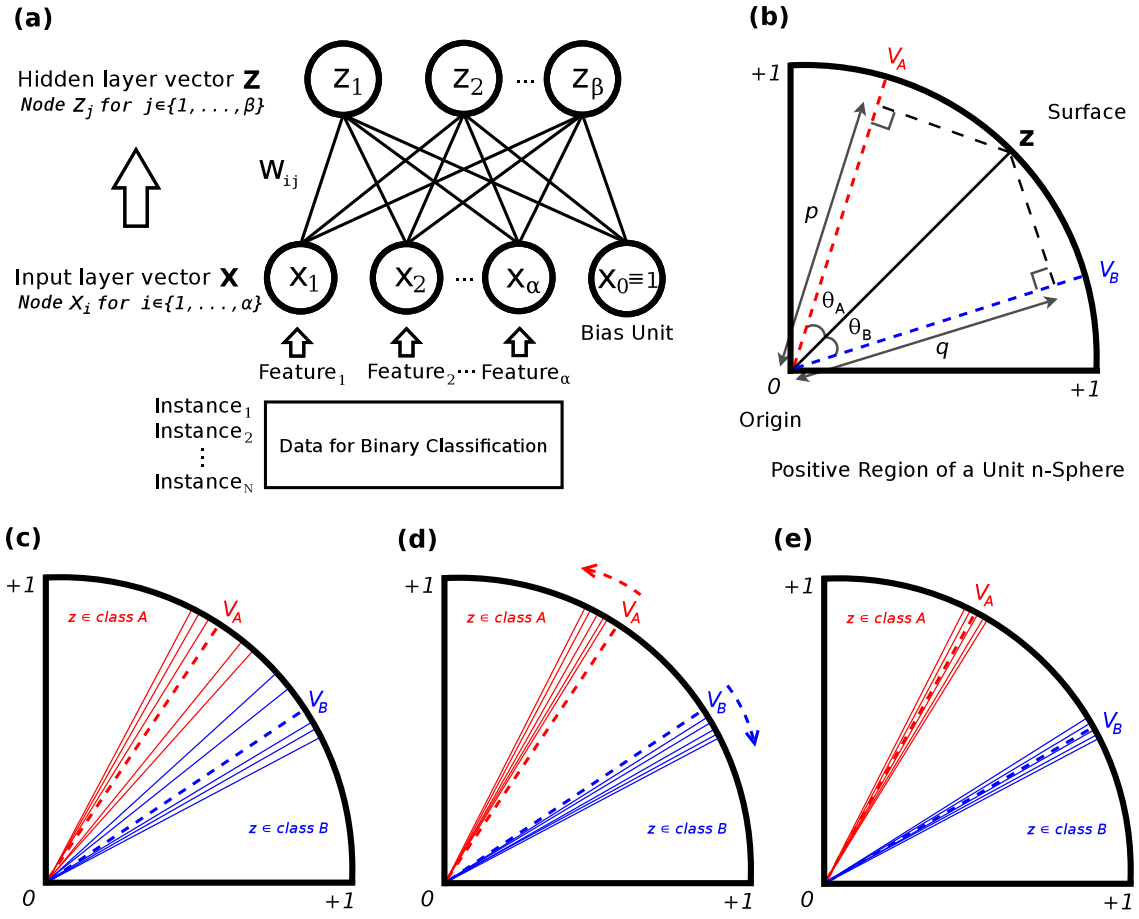

Figure 4.1: DCLN structure.
Revised from the original picture in our paper (©2015 IEEE).

The DCLN structure with $\alpha+1$ features at the input layer and β neurons at the hidden layer is depicted in Figure 4.1(a) in which the network is fully connected. Each hidden unit j takes the net input and convert using a nonlinear activation function:

$$z_j^g = g(\mathbf{w}_j^T \mathbf{x}) = g(x_0 w_{0j} + \sum_{i=1}^{\alpha} x_i w_{ij}), \quad j = 1, \ldots, \beta, \tag{4.1}$$

where $\mathbf{w}_j = (w_{0j}, w_{1j}, \ldots, w_{\alpha j})^T$ are weights connecting the hidden neuron j to all input

features in $\mathbf{x}$, and w_{0j} is the weight linking to the bias unit $x_0 \equiv +1$.

The nonlinear activation function $g(\cdot)$ being used for hidden neuron j takes the form of *logistic sigmoid* function shown as below:

$$g(t) = \frac{1}{1 + \exp(-\frac{1}{C}t)}, \tag{4.2}$$

where $0 < g(t) < 1$, and constant $\frac{1}{C}$ controls the slope of the function [45].

Once the input vector $\mathbf{x}$ is converted into a vector $\mathbf{z}^g$ at the hidden layer ($\forall \mathbf{z}^g \in \mathbb{R}^\beta$), we normalise each element z_j^g into z_j as follows:

$$z_j = \frac{z_j^g}{\sqrt{\sum_{k=1}^{\beta}(z_k^g)^2}} = \frac{z_j^g}{||\mathbf{z}^g||_2^2}, \quad j = 1, \ldots, \beta, \tag{4.3}$$

where $||\cdot||_2^2$ represents the ℓ_2 norm of a vector. Here, we end up with a normalised hidden layer vector $\mathbf{z}$ composing of z_j, and we have $||\mathbf{z}||_2^2 = 1$ and $0 \leq z_j \leq 1$. Because the ℓ_2 norm of the vector $\mathbf{z}$ is equal to 1 and each z_j is greater than 0, the normalised vector $\mathbf{z}$ is now a unit vector be part of the *positive region* of a unit n-sphere, linking the origin of the sphere to a surface point in that region. The idea is simply illustrated in Figure 4.1(b). So far, every input instance $\mathbf{x}$ in either class, can be converted into such unit vector $\mathbf{z}$: $\mathbf{x} \mapsto \mathbf{z}$.

Next, we calculate a *centroid* for the cluster of normalised $\mathbf{z}$ vectors of each class y. The two centroids are class representatives at the hidden layer. $\forall \mathbf{x} \mapsto \mathbf{z}$ in a class y, the centroid $\mathbf{v}_y$ is also a β-dimensional vector containing elements v_{yj} ($j = 1, \ldots, \beta$) which can be defined as follows:

$$v_{yj} = \frac{\sum_{n \in y} z_{nj}}{\sqrt{\sum_{k=1}^{\beta}(\sum_{n \in y} z_{nk})^2}}, \quad y \in \{\mathcal{A}, \mathcal{B}\}, \tag{4.4}$$

where v_{yj} and z_{nj} are indexed by j^{th} position at the hidden layer for the n^{th} instance in the class y. Since the two centroids are actually computed from the hidden neuron vectors, the norm $||\mathbf{v}||_2^2 = 1$ and range $0 \leq \mathbf{v_j} \leq 1$ also hold in this case. Here, $\mathbf{v}$ is a generic representation, either $\mathbf{v}_\mathcal{A}$ or $\mathbf{v}_\mathcal{B}$. Once the two centroids are obtained, each input instance

can be mapped to the hidden layer and then measure its distances to the two centroids by projecting to $\mathbf{v}_{\mathcal{A}}$ and $\mathbf{v}_{\mathcal{B}}$ respectively. Such projections can be calculated by simply taking the inner products between the vector and the centroids:

$$\langle \mathbf{z}, \mathbf{v}_{\mathcal{A}} \rangle = ||\mathbf{z}|| \cdot ||\mathbf{v}_{\mathcal{A}}|| \cdot \cos(\theta_A) = 1 \cdot \cos(\theta_A) = p,$$

$$\langle \mathbf{z}, \mathbf{v}_{\mathcal{B}} \rangle = ||\mathbf{z}|| \cdot ||\mathbf{v}_{\mathcal{B}}|| \cdot \cos(\theta_B) = 1 \cdot \cos(\theta_B) = q,$$

where θ_A and θ_B are the angles between the vector $\mathbf{z}$ and two centroids respectively (Figure 4.1(b)). Because the norm is zero for all vectors, taking a cosine function of the angles leads to the projections p and q respectively as the results of the two inner products.

Based on the geometric meaning of the projections, we can reshape an instance's memberships to the two classes as the posterior probabilities at the hidden layer. One simple way of defining it could be:

$$P(y = \mathcal{A}|\mathbf{z}) := \frac{p}{p+q}, \quad P(y = \mathcal{B}|\mathbf{z}) := 1 - P(y = \mathcal{A}|\mathbf{z}) = \frac{q}{p+q}, \tag{4.5}$$

ensuring that the two posterior probabilities of class membership sum to one.

Given an input instance, e.g. $\mathbf{x} \in \mathcal{A}$, its membership to the two classes are $P(y = \mathcal{A}|\mathbf{x} \in \mathcal{A}) = 1$ and $P(y = \mathcal{B}|\mathbf{x} \in \mathcal{A}) = 0$ respectively in the original input space. However, once it is mapped to the hidden layer and represented by the normalised unit vector $\mathbf{z}$, its membership to class $\mathcal{A}$ is redefined by equation 4.5 and is reduced below one at $\mathbf{z}$: $P(y = \mathcal{A}|\mathbf{z} \in \mathcal{A}) = p/(p+q) \leq 1$ (due to the mapping $\mathbf{x} \mapsto \mathbf{z}$ is guided by the weight matrix $\mathbf{w}$).

The objective here is to recover its membership probability back to one for class $\mathcal{A}$ at the hidden layer. The membership recovery will force the instance projecting more on its own centroid than projecting on the centroid of the opposite class. To maximise $P(y = \mathcal{A}|\mathbf{z}) = p/(p+q)$ in this example, the following is to be satisfied for recovery.

$$\max_{q/p} P(y = \mathcal{A}|\mathbf{z}) = \lim_{q/p \to 0} \frac{1}{1 + \frac{q}{p}} = 1 = P(y = \mathcal{A}|\mathbf{x} \in \mathcal{A}), \tag{4.6}$$

where $q/p \to 0$ in equation 4.6 implies possible scenarios:

For example, one could be $0 < q \ll p < 1$, representing $\mathbf{z} \in \mathcal{A}$ is projected to $\mathbf{v}_\mathcal{A}$ much more than the projection to $\mathbf{v}_\mathcal{B}$. In the ideal situation when p and q are almost equal to 1 and 0 respectively, the instance is very close to the centroid $\mathbf{v}_\mathcal{A}$, while it is nearly orthogonal to the centroid $\mathbf{v}_\mathcal{B}$ at the same time.

Another example for the given instance could be $q = 0$ and $0 < p < 1$. Consider, for the time being, a possible situation for instances in class $\mathcal{A}$. The subspace $\mathbb{M}_\mathcal{A} \in \mathbb{R}^\beta$ defined by $\forall \mathbf{z} \in \mathcal{A}$ is orthogonal to the subspace $\mathbb{M}_\mathcal{B} \in \mathbb{R}^\beta$ defined by $\forall \mathbf{z} \in \mathcal{B}$ with the properties as follows:

$$\mathbb{M}_\mathcal{B} = \{\forall \mathbf{z} \in \mathcal{B} | \langle \mathbf{z}, \mathbf{v}_\mathcal{B} \rangle = 1\} \subseteq \mathbb{M}_\mathcal{A}^\perp = \{\mathbf{z}^* \in \mathbb{R}^\beta | \langle \mathbf{z}^*, \mathbf{z} \rangle = 0, \forall \mathbf{z} \in \mathbb{M}_\mathcal{A}\}$$

and

$$\mathbb{M}_\mathcal{A} = \{\forall \mathbf{z} \in \mathcal{A} | 0 < \langle \mathbf{z}, \mathbf{v}_\mathcal{A} \rangle < 1 \wedge \neg \Box (\langle \mathbf{z}, \mathbf{v}_\mathcal{A} \rangle \approx 1)\}.$$

It implies that every $\mathbf{z} \in \mathcal{A}$ is orthogonal to every $\mathbf{z} \in \mathcal{B}$ (perfectly tight cluster, all vectors are equal in $\mathcal{B}$), but still with certain projection to its own centroid $\mathbf{v}_\mathcal{A}$. In this situation, the two distinct clusters can be formed to support an effective classification but there is no guarantee to generate a perfectly tight cluster for instances in class $\mathcal{A}$, even if their memberships are fully recovered to $\mathcal{A}$ at the hidden layer.

However in both scenarios, an instance is pushed away from its opposite centroid and generally gets closer to its own one (Figure 4.1(c)(d)) via the reconstruction of the class membership at the hidden layer. Lastly, in perfect situation, an optimal set of weights $\mathbf{w}^*$ can be found to guide the mapping of $\mathbf{x} \mapsto \mathbf{z}$ such that all instances of the same class receive a 100% membership of their own class and 0% of the opposite class, if the following equation receives its theoretical maximum N:

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \sum_{\mathcal{Y} \in \{\mathcal{A}, \mathcal{B}\}} \sum_{n \in \mathcal{Y}} \langle \mathbf{z}_n, \mathbf{v}^*_\mathcal{Y} - \mathbf{v}^*_{\neg \mathcal{Y}} \rangle.$$

If such solution exists and can be achieved in a way, then the common structural regularities within the same class are well extracted, while the structural differences between the two classes are maximally amplified at the hidden layer. At the same time, the noise

that is unrelated to the structural information in data is minimised.

However, it is not straight forward to directly implement that idea, and instead, we introduce the following two ways of maximising the class membership by either minimising the *cross-entropy* (CE) or the *squared error* (SE) loss function. Such optimisation methods provide pathways to learn a relatively practical mapping, and hence allowing opportunities to learn a desired set of weights as the solution. We begin with the cross-entropy method first.

Firstly, we employ an indicator function for an instance $\mathbf{x}_n$ as below:

$$\gamma_n = \gamma(\mathbf{x}_n) = \begin{cases} 1, & \text{if } \mathbf{x}_n \in \text{class } \mathcal{A} \\ 0, & \text{otherwise.} \end{cases}$$

Here, it follows a Bernoulli distribution:

$$\gamma_n | \mathbf{x}_n \mapsto \mathbf{z}_n \sim \text{Bernoulli}(P(y_n = \mathcal{A}|\mathbf{z}_n)),$$

and consequently the likelihood function given data $\mathcal{D}$ can be expressed as:

$$\mathcal{L}(\mathbf{w}|\mathcal{D}) = \prod_{n=1}^{N} P(y_n = A|\mathbf{z})^{\gamma_n} P(y_n = B|\mathbf{z})^{1-\gamma_n}, \tag{4.7}$$

where $P(y_n = B|\mathbf{z})^{1-\gamma_n} = (1 - P(y_n = A|\mathbf{z}))^{1-\gamma_n}$. For any instance in any case, one term must be equal to 1 in equation 4.7, either $P(y_n = A|\mathbf{z})^{\gamma_n}$ or $P(y_n = B|\mathbf{z})^{1-\gamma_n}$ because one indicator function must be 0, either γ_n or $1 - \gamma_n$. In such a case, the likelihood function reaches its maximum value 1 when $\forall P(y_n = A|\mathbf{z})^{\gamma_n} \equiv 1$ for $\gamma_n = 1$, and $\forall P(y_n = B|\mathbf{z})^{1-\gamma_n} \equiv 1$ for $\gamma_n = 0$. This means, if the likelihood function can be optimised to its maximum, the membership probability for the given instance will be recovered accordingly at the hidden layer.

As can be seen, the likelihood function needs to be maximised with respect to $P(y_n = A|\mathbf{z})$ only. A common way of maximising a likelihood function is to convert it into the

corresponding cross-entropy to minimise as follows:

$$\mathcal{E}(\mathbf{w}|\mathcal{D}) = -\sum_{n=1}^{N}\{\gamma_n \log P(y_n = \mathcal{A}|\mathbf{z}) + (1-\gamma_n)\log(1 - P(y_n = \mathcal{A}|\mathbf{z}))\}. \tag{4.8}$$

In this situation the loss function needs to be minimised with respect to $P(y_n = A|\mathbf{z})$, by which the function can be ultimately minimised with respect to the weights $\mathbf{w}$. As a general strategy, a gradient-based optimisation method can be employed to search for the optimal solution $\mathbf{w}^*$. Here, the widely used stochastic gradient descent (SGD) is applied in an *online learning* fashion (the weights are updated for each input presented to the network) to minimise the loss function, aiming at local minima:

$$\exists_{\mathbf{w}^*} \nabla \mathcal{E}(\mathbf{w}|\mathcal{D})|_{\mathbf{w}=\mathbf{w}^*} = 0.$$

For each *epoch*, all instances are mapped to the hidden layer to calculate $\mathbf{v}_\mathcal{A}$ and $\mathbf{v}_\mathcal{B}$, and then each mapped instance $\mathbf{z}$ is randomly selected by the training procedure once only to perform an update to the weights at that epoch. For each presented instance, the weights $\mathbf{w}$ are modified in the direction that is opposite to the gradient by a certain amount as: $\Delta w_{ij} = -\eta \frac{\partial \mathcal{E}}{\partial w_{ij}}$, where η is the learning rate, and the weights update Δw_{ij} can be calculated by the chain rule of partial derivatives, shown as below:

$$\Delta w_{ij} = -\eta \frac{\partial \mathcal{E}(\mathbf{w}|\mathcal{D})}{\partial w_{ij}} = -\eta \frac{\partial \mathcal{E}(\mathbf{w}|\mathcal{D})}{\partial P(y = \mathcal{A}|\mathbf{z})} \cdot \frac{\partial P(y = \mathcal{A}|\mathbf{z})}{\partial z_j} \cdot \frac{\partial z_j}{\partial z_j^g} \cdot \frac{\partial z_j^g}{\partial w_{ij}}. \tag{4.9}$$

With sufficient number of training epochs, the minimisation procedure is completed. To solve Δw_{ij} for each update, we show how the four elementary terms in equation 4.9 can be derived one by one as follows. The first term is easy to be obtained. Simply by regarding equation 4.8 as a function of $P(y = \mathcal{A}|\mathbf{z})$:

$$-\eta \frac{\partial \mathcal{E}(\mathbf{w}|\mathcal{D})}{\partial P(y = \mathcal{A}|\mathbf{z})} = \eta \left(\frac{\gamma_n}{P(y = \mathcal{A}|\mathbf{z})} - \frac{1-\gamma_n}{1 - P(y = \mathcal{A}|\mathbf{z})} \right). \tag{4.10}$$

Recall from equation 4.5 that $P(y = \mathcal{A}|\mathbf{z})$ is composed of p and q, which represent the inner products of the instance to the two centroids $\mathbf{v}_\mathcal{A}$ and $\mathbf{v}_\mathcal{B}$ respectively. In order to

explicitly express the inner product as a function of z_j for the hidden unit j, we rewrite equation 4.5 in the following form:

$$P(y = \mathcal{A}|\mathbf{z}) = \frac{p}{p+q} = \frac{U(z_j)}{V(z_j)}, \tag{4.11}$$

where

$$U(z_j) = p = \langle \mathbf{z}, \mathbf{v}_{\mathcal{A}} \rangle = \frac{\sum_{k=1}^{\beta}(z_k \sum_{n\in\mathcal{A}} z_{nk})}{\sqrt{\sum_{k=1}^{\beta}(\sum_{n\in\mathcal{A}} z_{nk})^2}} = \frac{z_j \sum_{n\in\mathcal{A}} z_{nj} + \sum_{k'=1}^{\beta}(z_{k'} \sum_{n\in\mathcal{A}} z_{nk'})}{\sqrt{\sum_{k=1}^{\beta}(\sum_{n\in\mathcal{A}} z_{nk})^2}} \tag{4.12}$$

for $k^{'} \neq j$, and

$$V(z_j) = p + q = \langle \mathbf{z}, \mathbf{v}_{\mathcal{A}} \rangle + \langle \mathbf{z}, \mathbf{v}_{\mathcal{B}} \rangle = \sum_{\mathcal{Y}\in\{\mathcal{A},\mathcal{B}\}} \frac{\sum_{k=1}^{\beta}(z_k \sum_{n\in\mathcal{Y}} z_{nk})}{\sqrt{\sum_{k=1}^{\beta}(\sum_{n\in\mathcal{Y}} z_{nk})^2}}. \tag{4.13}$$

Because U and V are made up of z_j (defined by equation 4.3), the $P(y = \mathcal{A}|\mathbf{z})$ of equation 4.11 is also expressed as a function of z_j to solve equation 4.10.

Next, we show how the second term in equation 4.9 can be obtained. This term requires calculating the derivatives of U and V as part of the solution:

$$\frac{\partial P(y = \mathcal{A}|\mathbf{z})}{\partial z_j} = \frac{\partial}{\partial z_j}\left(\frac{U(z_j)}{V(z_j)}\right) = \left(\frac{U}{V}\right)^{'} = \frac{U'V - V'U}{V^2}, \tag{4.14}$$

where $U^{'}$ and $V^{'}$ denote the derivatives with respect to z_j for U and V respectively. At a given epoch two centroids are temporarily fixed, and in equations 4.12 and 4.13 we can treat z_j as the only variable, leaving centroid elements and other components as constants for that epoch. Consequently, two derivatives can be expressed respectively in the following forms:

$$U' = \frac{dU(z_j)}{dz_j} = \frac{\sum_{n\in\mathcal{A}} z_{nj}}{\sqrt{\sum_{k=1}^{\beta}(\sum_{n\in\mathcal{A}} z_{nk})^2}} = v_{\mathcal{A}j} \tag{4.15}$$

and

$$V' = \frac{dV(z_j)}{dz_j} = \frac{\sum_{n\in\mathcal{A}} z_{nj}}{\sqrt{\sum_{k=1}^{\beta}(\sum_{n\in\mathcal{A}} z_{nk})^2}} + \frac{\sum_{n\in\mathcal{B}} z_{nj}}{\sqrt{\sum_{k=1}^{\beta}(\sum_{n\in\mathcal{B}} z_{nk})^2}} = v_{\mathcal{A}j} + v_{\mathcal{B}j}. \tag{4.16}$$

Therefore, the second partial derivative term in equation 4.9 is resolved by substituting equations 4.12, 4.13, 4.15 and 4.16 into 4.14.

Next, we focus on the third term $\frac{\partial z_j}{\partial z_j^g}$ in equation 4.9. We first directly present the solution as follows and then show the proof of it.

$$\frac{\partial z_j}{\partial z_j^g} = \sum_{k'=1}^{\beta} (z_{k'}^g)^2 (\sum_{k=1}^{\beta} (z_k^g)^2)^{-\frac{3}{2}} \quad \text{for } k' \neq j. \tag{4.17}$$

Proof of equation 4.17. *We can rewrite equation 4.3 into*

$$z_j = 1/\sqrt{\frac{\sum_{k=1}^{\beta}(z_k^g)^2}{(z_j^g)^2}} = 1/\sqrt{1 + \frac{\sum_{k'=1}^{\beta}(z_{k'}^g)^2}{(z_j^g)^2}} \quad \textit{for } k' \neq j.$$

Now let

$$T = 1 + \frac{\sum_{k'=1}^{\beta}(z_{k'}^g)^2}{(z_j^g)^2}$$

and we can conveniently compute

$$\frac{dz_j}{dT} = -\frac{1}{2}T^{-\frac{3}{2}} = -\frac{1}{2}(1 + \frac{\sum_{k'=1}^{\beta}(z_{k'}^g)^2}{(z_j^g)^2})^{-\frac{3}{2}} = -\frac{1}{2}(\frac{\sum_{k=1}^{\beta}(z_k^g)^2}{(z_j^g)^2})^{-\frac{3}{2}} \tag{4.18}$$

as well as

$$\frac{dT}{dz_j^g} = -2(z_j^g)^{-3} \sum_{k'=1}^{\beta} (z_{k'}^g)^2 \quad \textit{for } k' \neq j. \tag{4.19}$$

Equations 4.18 and 4.19 consequently give us the result that

$$\frac{\partial z_j}{\partial z_j^g} = \frac{dz_j}{dT} \cdot \frac{dT}{dz_j^g} = (z_j^g)^{-3} \sum_{k'=1}^{\beta} (z_{k'}^g)^2 (\frac{\sum_{k=1}^{\beta}(z_k^g)^2}{(z_j^g)^2})^{-\frac{3}{2}} = \sum_{k'=1}^{\beta} (z_{k'}^g)^2 (\sum_{k=1}^{\beta}(z_k^g)^2)^{-\frac{3}{2}} \quad \textit{for } k' \neq j. \tag{4.20}$$

□

Lastly, the final term $\frac{\partial z_j^g}{\partial w_{ij}}$ in equation 4.9 is dependent on the input units. In general it takes the following form:

$$\frac{\partial z_j^g}{\partial w_{ij}} = \frac{1}{C} \cdot x_i \cdot g(\sum_{i=1}^{\alpha} x_i w_{ij})(1 - g(\sum_{i=1}^{\alpha} x_i w_{ij})) = \frac{1}{C} x_i z_j^g (1 - z_j^g). \tag{4.21}$$

So far we have explicitly derived all terms required by equation 4.9. Therefore, we can update $\mathbf{w}$ for every instance presented to the network by Δw_{ij} to minimise the cross-entropy function described by equation 4.8. As can be seen from equation 4.9, the minimisation of the cross-entropy loss with respect to the weights can be decomposed into four terms, and this idea can be easily extended into a more flexible form. An alternative loss function $\mathcal{E}$, regarded as a function of $P(y = \mathcal{A}|\mathbf{z})$ (we denote $P_{\mathcal{A}}$ to it in the following equation for simplicity), can also be used when:

$$\exists \mathcal{E}^{'} := f(P_{\mathcal{A}}) \wedge \inf_{\neg \forall P_{\mathcal{A}} = \gamma_n} \mathcal{E}(P_{\mathcal{A}}) = \mathcal{E}(P_{\mathcal{A}})|_{\forall P_{\mathcal{A}} = \gamma_n}, \tag{4.22}$$

requiring the first term in equation 4.9 to be replaced by $-\eta \cdot f(P_{\mathcal{A}})$, leaving the rest terms same as before. Based on this, we show an alternative loss function using squared error, which generally takes the following form:

$$\mathcal{E}(\mathbf{w}|\mathcal{D}) = \frac{1}{2}\sum_{n=1}^{N} ||\gamma_n - P(y_n = \mathcal{A}|\mathbf{z})||^2 = \frac{1}{2}\sum_{n=1}^{N} ||1 - \gamma_n - P(y_n = \mathcal{B}|\mathbf{z})||^2. \tag{4.23}$$

Because both squared terms in equation 4.23 are equal or greater than zero, the function reaches its minimum value zero when $\forall P(y_n = A|\mathbf{z}) \equiv 1$ for $\gamma_n = 1$, and $\forall P(y_n = B|\mathbf{z}) \equiv 1$ for $\gamma_n = 0$. Therefore, the minimisation of such function also leads to the membership recovery at the hidden layer.

Similar to cross-entropy method, SGD is also applied here searching for the minimum. Same as before, the weights $\mathbf{w}$ are updated for every instance by Δw_{ij} whose chain rule of partial derivatives are almost the same as equation 4.9 except for the first term to be replaced by the following term:

$$-\eta \frac{\partial \mathcal{E}(\mathbf{w}|\mathcal{D})}{\partial P(y = \mathcal{A}|\mathbf{z})} = \eta(\gamma_n - P(y_n = \mathcal{A}|\mathbf{z})).$$

The three remaining terms are still expressed by equations 4.14, 4.17 and 4.21 respectively.

4.2.2 Algorithm

Putting all components together, the DCLN algorithm details are given here. For a train-

Algorithm 3 Training DCLN with cross entropy (CE) and squared error (SE) methods.

Require: Input data $\mathcal{D} = \{\mathbf{x}_n, y_n\}_{n=1}^{N}$, $y \in \{\mathcal{A}, \mathcal{B}\}$.

Set loss function, either CE or SE.
Set hyperparameters: hidden layer dimension β; learning rate η; sigmoid slope factor C; and maximum *epoch*.
Randomly initiate $\mathbf{w}$ in a proper range: e.g. $w_{ij} \in [-0.5, 0.5]$, where $i \in [0, \ldots, \alpha]$ for input layer and $j \in [1, \ldots, \beta]$ for hidden layer.
Initialise $\mathbf{w}^*$, $\mathbf{v}_{\mathcal{A}}^*$ and $\mathbf{v}_{\mathcal{B}}^* \leftarrow \varnothing$; $min \leftarrow 1$.
repeat
  for each instance $\mathbf{x}_n$, where $n \in [1, \ldots, N]$ **do**
    $\mathbf{z}_n^g$: $z_{nj}^g \leftarrow 1/(1 + e^{-\frac{1}{C}(\mathbf{w}_j^T \mathbf{x}_n)})$, for $j \in [1, \ldots, \beta]$.
    $\mathbf{z}_n$: $z_{nj} \leftarrow z_{nj}^g / \sqrt{\sum_{k=1}^{\beta} (z_{nk}^g)^2}$, for $j \in [1, \ldots, \beta]$.
  end for
  for $y \in \{\mathcal{A}, \mathcal{B}\}$ **do**
    $\mathbf{v}_y$: $v_{yj} \leftarrow (\sum_{n \in y} z_{nj}) / \sqrt{\sum_{k=1}^{\beta} (\sum_{n \in y} z_{nk})^2}$, for $j \in [1, \ldots, \beta]$.
  end for
  if $\langle \mathbf{v}_{\mathcal{A}}, \mathbf{v}_{\mathcal{B}} \rangle < min$ **then**
    $\mathbf{w}^* \leftarrow \mathbf{w}$; $\mathbf{v}_{\mathcal{A}}^* \leftarrow \mathbf{v}_{\mathcal{A}}$; $\mathbf{v}_{\mathcal{B}}^* \leftarrow \mathbf{v}_{\mathcal{B}}$; $min \leftarrow \langle \mathbf{v}_{\mathcal{A}}, \mathbf{v}_{\mathcal{B}} \rangle$.
  end if
  for every instance $\mathbf{x}_n$ in random order **do**
    $U \leftarrow \langle \mathbf{z}_n, \mathbf{v}_{\mathcal{A}} \rangle$; $V \leftarrow \langle \mathbf{z}_n, \mathbf{v}_{\mathcal{A}} + \mathbf{v}_{\mathcal{B}} \rangle$.
    for $j \in [1, \ldots, \beta]$ **do**
      $U^{'} \leftarrow v_{\mathcal{A}j}$; $V^{'} \leftarrow v_{\mathcal{A}j} + v_{\mathcal{B}j}$.
      for $i \in [0, \ldots, \alpha]$ **do**
        $\Delta \mathbf{w} \leftarrow (U^{'}V - V^{'}U)/V^2 \cdot \sum_{k^{'}=1}^{\beta} (z_{nk^{'}}^g)^2 \cdot (\sum_{k=1}^{\beta} (z_{nk}^g)^2)^{-\frac{3}{2}} \cdot \frac{1}{C} \cdot x_{ni} \cdot z_{nj}^g (1 - z_{nj}^g)$, for $k^{'} \neq j$.
        if $\mathbf{x}_n \in \mathcal{A}$ **then**
          $w_{ij}^{(CE)} \leftarrow w_{ij}^{(CE)} + \eta \cdot V/U \cdot \Delta \mathbf{w}$.
          $w_{ij}^{(SE)} \leftarrow w_{ij}^{(SE)} + \eta \cdot (1 - U/V) \cdot \Delta \mathbf{w}$.
        else
          $w_{ij}^{(CE)} \leftarrow w_{ij}^{(CE)} - \eta/(1 - U/V) \cdot \Delta \mathbf{w}$.
          $w_{ij}^{(SE)} \leftarrow w_{ij}^{(SE)} - \eta \cdot U/V \cdot \Delta \mathbf{w}$.
        end if
      end for
    end for
  end for
until epoch=h.
Output $\mathbf{w}^*$, $\mathbf{v}_{\mathcal{A}}^*$ and $\mathbf{v}_{\mathcal{B}}^*$.

Slightly revised from the original algorithm in our paper (©2015 IEEE).

ing dataset $\mathcal{D}$, the weights $\mathbf{w}^0$ are randomly initialised. At the initial epoch, each input vector $\mathbf{x}$ is mapped to the hidden layer by equation 4.2 and then normalised as $\mathbf{z}$ by equation 4.3. Once all instances of the two classes are converted into the unit vectors, the two centroids are calculated by equation 4.4 and temporarily fixed for the current epoch (Figure 4.1(c)).

With calculated centroids, each unit vector $\mathbf{z}$ is then randomly given to the weights-updating process. Since z^g, z, U, V, $U^{'}$, $V^{'}$, p, q and $P(y_n = A|\mathbf{z})$ are all clear for that vector, we can update the weights $\mathbf{w}^n \leftarrow \mathbf{w}^{n-1} + \Delta\mathbf{w}$ using equation 4.9. Generally, instances move away from their opposite class centroid and get close to their own one (Figure 4.1(d)). This recovers the membership probability to some extent at the hidden layer as discussed earlier. After that, the training process proceeds to the next epoch, where two centroids are re-calculated and updated to the new positions (Figure 4.1(e)). The whole procedure is repeated until the pre-defined number of epochs is reached. The model training of DCLN is detailed in Algorithm 3.

During the training, it is expected to see that two centroids are moving in the opposite way of each other. Since they are class representatives, the increased distance between them (decreased inner product $\langle \mathbf{v}_{\mathcal{A}}, \mathbf{v}_{\mathcal{B}} \rangle$) implies that the structural differences between the two categories are gradually learned by the algorithm. Once the solutions $\mathbf{w}^*$, ${\mathbf{v}_{\mathcal{A}}}^*$ and ${\mathbf{v}_{\mathcal{B}}}^*$ are determined, the binary classification can be done by simply mapping an unknown instance to the hidden layer $\mathbf{x} \mapsto \mathbf{z}$ using $\mathbf{w}^*$, and calculate $\langle \mathbf{z}, \mathbf{v}^*_{\mathcal{A}} \rangle$ and $\langle \mathbf{z}, \mathbf{v}^*_{\mathcal{B}} \rangle$ to determine the closest class centroid for the test instance.

One thing to note here is that for very imbalanced data, assuming $n_1 = 10$ instances for $\mathcal{A}$ and $n_2 = 1000$ instances for $\mathcal{B}$, instances in class $\mathcal{A}$ have much lower chances of getting into the update of $\mathbf{w}$. The update is dominated by instances in class $\mathcal{B}$, and hence the update is biased to the movement of $\mathbf{v}_{\mathcal{B}}$. To balance this, we can simply manipulate the learning rate by using an "adjusted" learning rate η^*, instead of η, differently to the two classes. Here, in Algorithm 3, we may use $\eta^*_{\mathcal{B}} = \eta \times n_1/n_2$ for $\mathbf{x} \in \mathcal{B}$ if $n_1 < n_2$ when updating $\mathbf{w}$. And symmetrically, use $\eta^*_{\mathcal{A}} = \eta \times n_2/n_1$ for $\mathbf{x} \in \mathcal{A}$ if $n_2 < n_1$. This is a simple idea of *slowing down* the update for the class with too many samples. The learning rate is one of the most important parameters and deserves more investigation in future.

The performance of DCLN is subject to a number of factors such as the number of hidden neurons β; learning rate η; sigmoid function slope factor C; and total number of epochs for the training procedure:

- Hidden neurons β: Sufficient number of hidden neurons are required for accommodating a complex problem, but the performance may not necessarily improve with an excessive amount of hidden neurons. Also, the computation time increases linearly with the number of neurons being used.
- Learning rate η: Controls the step of weights update and consequently affects the speed of convergence. Akin to other neural network models, a too small or overly large learning rate may also be accountable for slow convergence or unstable model training for DCLN.
- Sigmoid slope factor C: Controls the flatness/sharpness of the sigmoid function and is important for feature learning and prediction performance.
- Epochs: Sufficient number of cycles of training are necessary for extracting the structural information from the input data, but over-training may lead to an overfitting to the training data.

Apart from these factors, the model performance is also affected by the randomly initialised weights. With properly specified lower and upper bounds of an initialisation range, the model can achieve its optimal performance for a given dataset. Throughout the remaining part of this thesis, DCLN is extensively used in various simulation tests and real studies, with major hyperparameter settings specified for the experiments to demonstrate the idea. All other minor settings, optimisations and implementation details are skipped for simplicity. Experiments are performed on Linux system only.

4.3 Nonlinear Approximation Tests

The first test of DCLN is to check whether it is applicable to simple nonlinear problems. To test the capacity, we create four nonlinearly separable synthetic datasets. Both training and test sets contain $\sim$ 2000 samples for each study, and each category receives half of the samples. For each instance, there are two features x and y representing the horizontal

and vertical axes respectively. The instances (2D points) in red and blue belong to two different classes. Four studies are illustrated in Figure 4.2.

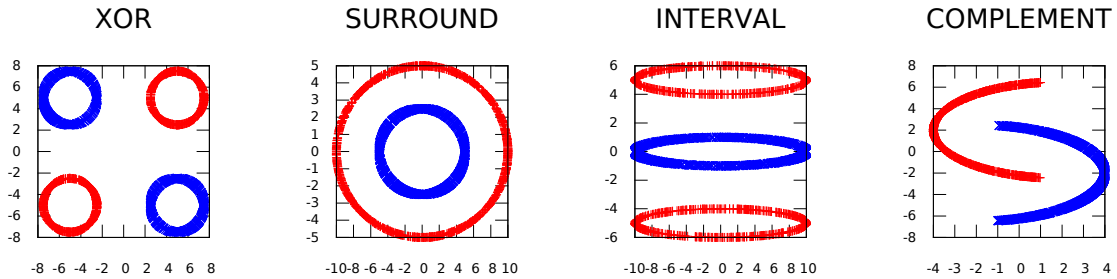

Figure 4.2: Nonlinearly separable data.
(©2015 IEEE)

Both DCLN CE and SE methods are applied with the following setting: $\beta = 200$, $\eta = 0.1$, $C = 0.1$ and $epoch = 400$, with additional settings optimised for these studies. In total, there are 500 repeated runs performed to calculate the average prediction performance for each study.

As a starting point, determining a proper random initialisation range of weights is important for DCLN. Here, we perform a comprehensive experiment to give a general idea of how well the proposed method works on nonlinear scenarios. Firstly, we test whether the model is tolerated by different ranges. Given the input dimension of three (raw features $\alpha = 2$ plus 1 bias unit) and $\beta = 200$ hidden nodes, the total number of weight connections are $3 \times 200 = 600$. All 600 weights are initialised randomly within a predefined range specified by its lower and upper bounds, and we test the following choices: $[\text{lower}, \text{upper}] \in \{[-0.1, 0.1], [-0.2, 0.2], \ldots, [-0.9, 0.9]\}$ for each study. The corresponding balanced accuracy results are shown for these ranges in Figure 4.3.

As can be seen from the plot, DCLN generally works well on all four studies under different range settings. For each study, using either CE or SE method, at least one or more settings can give a prediction score that is close to 100% of balanced accuracy. The easiest case to tell is *INTERVAL* for which the performance reaches 100% for all settings with both optimisation methods. For *XOR*, the performance generally peaked around $[-0.3, +0.3]$ for CE method. For SE method, on the other hand, the performance gradually decreased after $[-0.4, +0.4]$. For *SURROUND*, the CE method gives nearly 100%

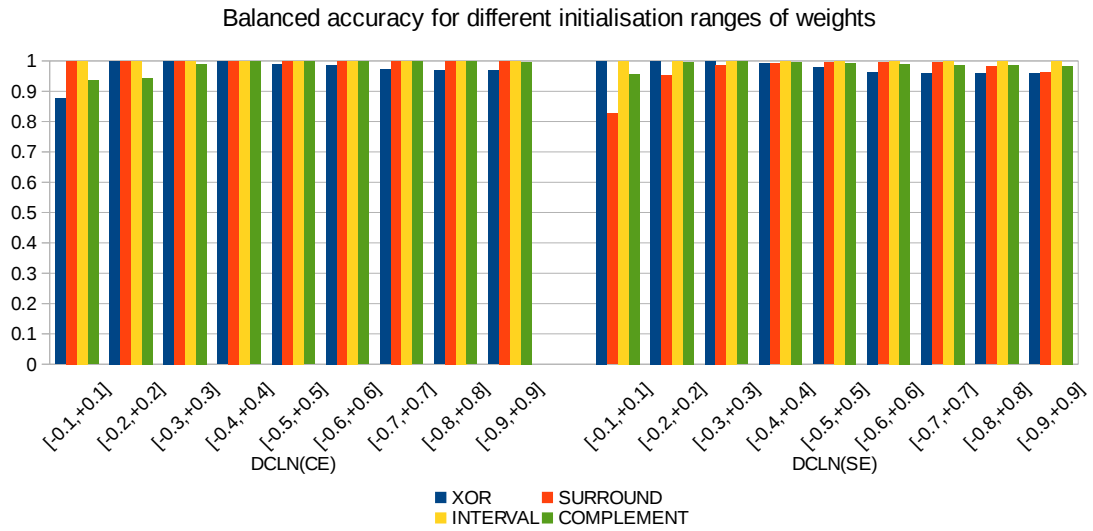

Figure 4.3: Prediction performance of various initialisation ranges.

balanced accuracy for all settings, while the SE method peaks around $[-0.5, +0.5]$. Lastly, for *COMPLEMENT*, the CE method reached to the top after $[-0.4, +0.4]$, and SE method also report quality results except for some degradations appeared in settings with too narrow or too broad ranges.

Given the determined initialisation range, we perform the next experiment to check whether the two centroids move away from each other as the training proceeds. The inner product of the two centroids $\langle \mathbf{v}_{\mathcal{A}}, \mathbf{v}_{\mathcal{B}} \rangle$ is recorded for every epoch. At the same time, the prediction on the test data is also performed at each epoch. Again, total 500 trials are performed for averaging the results, and the recorded numbers are illustrated in Figure 4.4 and 4.5 respectively.

As can be seen from Figure 4.4, the inner product decreases as the training proceeds for all studies. Therefore, the actual behaviour of the model agrees with what was expected by the theory. The structure differences of the two classes are gradually learned during the training and two clusters are indeed moving apart from each other. However, different methods converged to different levels of separation at their respective speed. For example, studies using CE generally converged more quickly than the corresponding SE methods. In most cases, CE method reaches to a lower level than SE method except for *SURROUND* study in which the SE method converged slower than CE but it arrived at a lower level in the end. On the other hand, different studies show comparable pat-

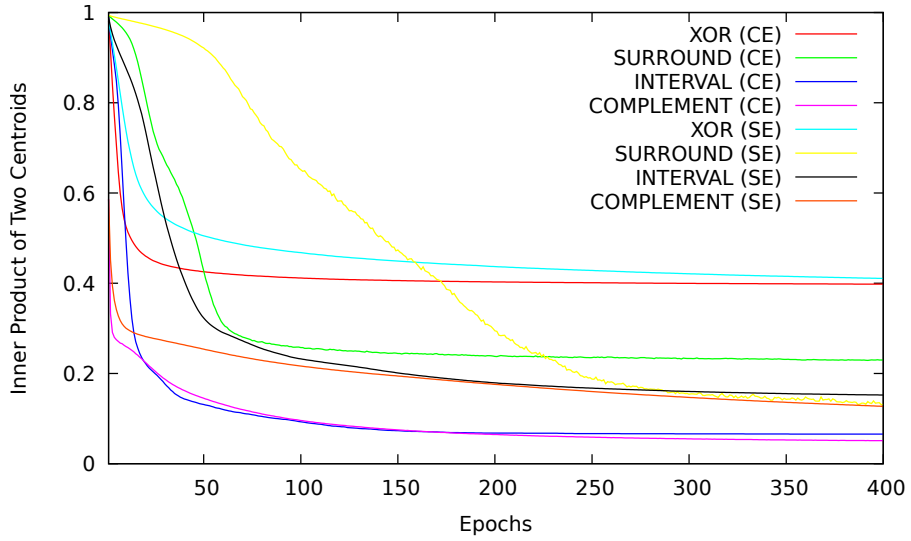

Figure 4.4: $\langle \mathbf{v}_{\mathcal{A}}, \mathbf{v}_{\mathcal{B}} \rangle$ at each training epoch.

terns as well. For example, *INTERVAL* and *COMPLEMENT* give curves that are similar in convergence speed and minimum after $\sim$ 100 epochs for both optimisation methods.

Next, we investigate the corresponding prediction performance along with the training process. As can be seen from Figure 4.5, all studies end up with nearly 100% balanced accuracy by the end of the training, regardless of their relative convergence levels of inner products described previously. This implies that even if the inner product is not minimised to the lowest level, the two class clusters are already well separated at the hidden layer to support an effective classification.

Apart from that, there are still some noticeable trends. Similar to the behaviour shown by the inner product plot, CE methods generally arrive at the maximum performance faster than corresponding SE methods. The slowly converged *SURROUND (SE)*, for example, reached to its maximum performance at a late stage of training.

In summary, these studies constitute a preliminary demonstration of the nonlinear approximation ability of DCLN. An immediate question here is how well the proposed method is able to generalise to the points that are largely deviated from the original training data. Although a separate testing set is created and used for each study in the previous experiment showing the prediction performance, those data points are randomly generated from the same signal region of the training data.

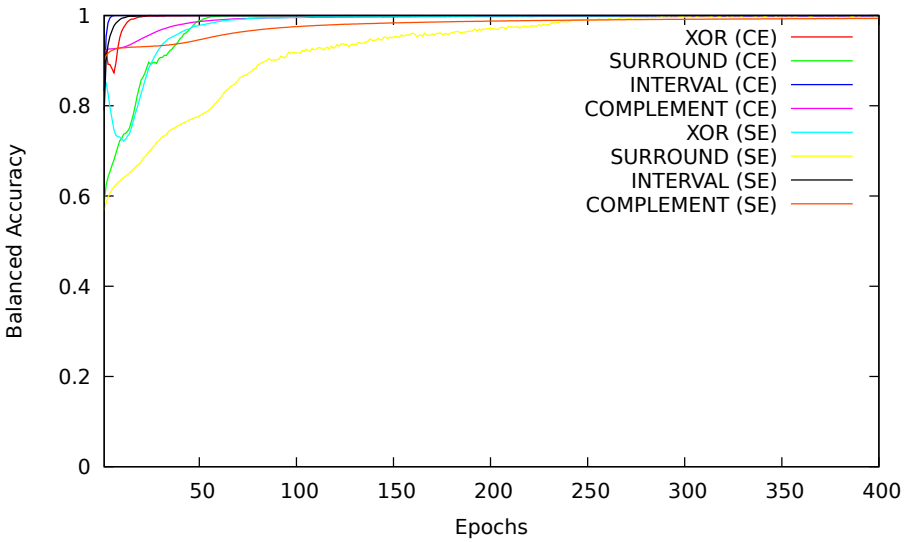

Figure 4.5: Prediction performance at each epoch.

In response, we are interested in seeing how the learned model labels each random point in the entire plotting area. More specifically, we generate a set of random points that are uniformly distributed over the whole area of the plot and see how those points are classified by DCLN, showing the decision boundary for each study.

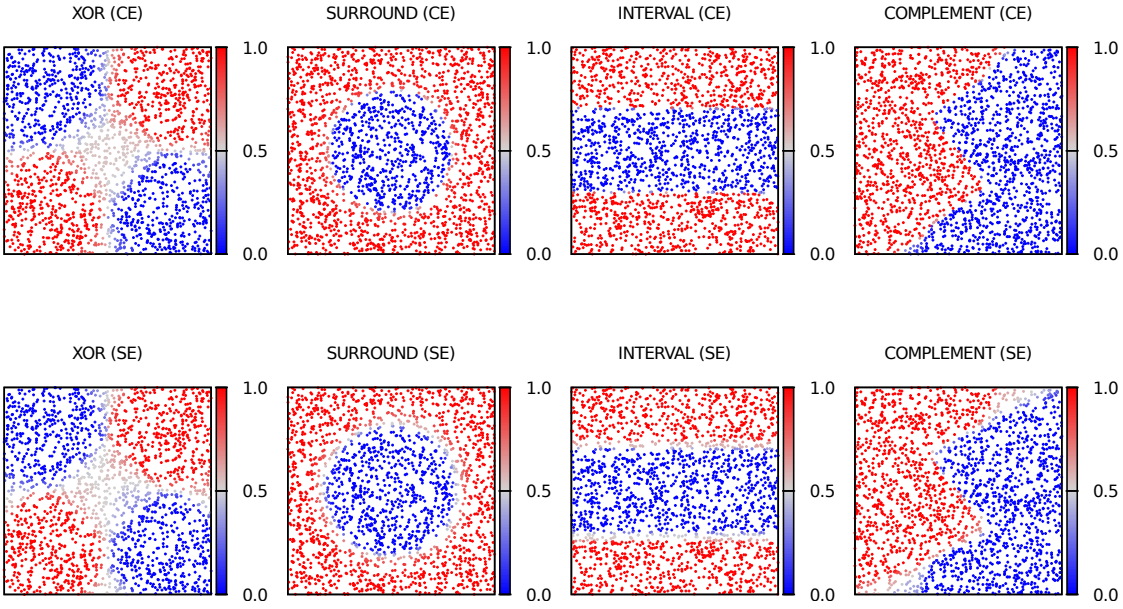

Figure 4.6: Nonlinear decision boundary for each study.

To test the model, 1000 such random points are generated as the noise set across the entire plotting area for each class of each study. The model is trained on the previous training set and applied to this noise set, labeling each point with the predicted class. Total 500 experiments are performed, and we investigate the possibility of a given random point to be assigned to class $\mathcal{A}$ by calculating how many times such point is labeled as $\mathcal{A}$ out of 500 trials. Such proportion value $P(\mathcal{A})$ is plotted for each random point in Figure 4.6, ranging from zero to one. Apparently, the point is shown as $\mathcal{A}$ (red) when $P(\mathcal{A}) > 0.5$, and $\mathcal{B}$ (blue) otherwise.

From the plot, it is clear that a correct decision boundary is learned for each study. For example, the majority of blue points in *XOR* study cover the 'top-left' and 'bottom-right' area for class $\mathcal{B}$, while the red points for class $\mathcal{A}$ take two complementary regions. This demonstrates that the derived model is correctly generalised to broader regions that contain the original data points. For the points that are lying on the decision boundaries, their values are approaching 0.5 correspondingly. This capacity is also shown by other studies, for which the regions rather than just the training data are realised by DCLN. A slight difference between CE and SE methods here is the learned CE boundary being relatively more clear than SE for the presented four studies. This may be due to the hyperparamters that are not optimised or because of the limitation of the method itself. We leave such investigation into future work.

In summary, DCLN correctly characterised the critical decision regions of these nonlinear problems to perform proper classifications for the test points. The learned regions show regularities, well accommodate the original points, and reasonably labeled those ones that are largely deviated from the training data.

4.4 Classifications with Nonlinear Interactions

Before showing how the high-level robust features can be selected by DCLN in later sections, we first give a preliminary exploration of its classification ability for nonlinear interaction studies. Although the objective is not proposing yet another state-of-the-art classification algorithm, the learned features by DCLN will not be convinced as effective

predictors without the support of a sufficiently well classification performance.

In this section, we benchmark DCLN against the SVM method, which has demonstrated its popularity for a broad range of applications in machine learning field. Both second and third order interactions are tested here using simulation datasets. The DCLN method closely follows the SVM to report high quality classification results. As other neural-net methods, DCLN is also a model-free method but it doesn't incur an extra overhead such as kernel selection required by SVM.

The first batch of tests is based on the four nonlinear patterns illustrated earlier in Table 2.4. The simulation data are generated for each of the four studies. Each category, either Case or Control, receives total 1000 samples and then they are distributed to the compartments of the contingency table according to the counts allocation specified in the table. The table is derived by the full penetrance function, representing ideal scenarios. In real world, this is less likely the case for most scenarios. Thus, we add various levels of random noise to the data for simulations. For each trial, we randomly select a proportion of samples and replace them with the random numbers $\{0,1,2\}$. We take the proportions of 0.02, 0.04, 0.2 and 0.4 for the total 1000 samples of each category to randomise. Therefore, we end up with total 4 (nonlinear interaction patterns) $\times$ 4 (noise level) $=$ 16 nonlinear datasets containing noise at different levels.

Next, for each trial, half of the samples are randomly allocated to the training set, leaving the rest as the test set. Within the training set, we use cross-validations for tuning the hyperparameters. For each cross-validation run, 75% samples are randomly selected from the training set for model fitting purpose, while the rest 25% are reserved for validation. The model is trained with each hyperparameter combination on the model-fitting set and then tested on the validation set. Total 50$\times$ cross-validations are performed and the best performing hyperparameter setting is determined as the one with the minimum average validation error over 50 trials. Once the best one is selected, it is used to train the model on the current training set and then predict on the test set to record a prediction score for the current trial. Above procedure is repeated for total 500 trials and the average prediction result is calculated.

For benchmarking different methods, we downloaded the widely-used SVM pro-

gram LIBSVM [16] version 3.24, and tested three popular kernels such as radial basis function (RBF), Polynomial and Linear kernels. For RBF kernel, a grid search for combinations of $C \in \{2^{-5}, 2^{-3}, 2^{+3}, 2^{+5}\}$ and $\gamma \in \{2^{-5}, 2^{-3}, 2^{+3}, 2^{+5}\}$ is performed. For Polynomial kernel, we tested $\{2, 3, 4, 5, 6\}$ degrees of polynomials. Lastly for linear kernel, we search through $C \in \{2^{-5}, 2^{-3}, 2^{+3}, 2^{+5}\}$. For DCLN, both CE and SE methods are tested in the experiment with the following hyperparameters search space: $\beta = 200$, $\eta \in \{0.01, 0.1, 1, 10\}$ and $C \in \{0.01, 0.1, 1, 10\}$. For simplicity, the weights are initialised in $[-0.5, +0.5]$ with other settings optimised for these studies. All hyperparameter combinations run for fixed 400 epochs.

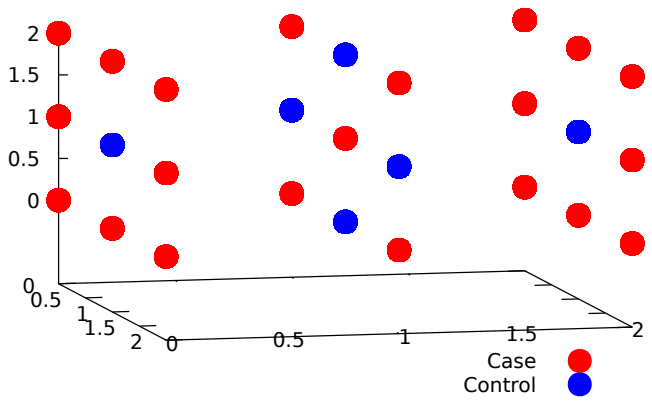

Figure 4.7: Third order interaction pattern I_J_K.

In addition to the pairwise interactions, we are also interested in examining the model's performance for third order interactions. Three interaction patterns are illustrated in Figures 4.7, 4.8 and 4.9 respectively, in which the samples of a particular class are evenly distributed to the cells of that class. Previously, 1000 instances are allocated to each class in a 3×3 contingency table of a binary interaction. Here, we increase to 3000 for a $3 \times 3 \times 3$ table of a ternary interaction. To reduce the extra computational cost incurred by the increased sample size, the total 400 epochs are cut down to 200 for these three studies.

To speed up the data processing, an Intel CPU cluster with Linux system is used for computing. The preliminary results are shown in Figure 4.10. As can be seen from

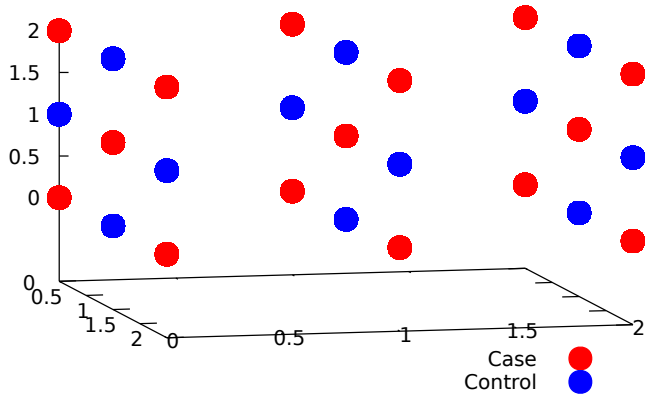

Figure 4.8: Third order interaction pattern L_M_N.

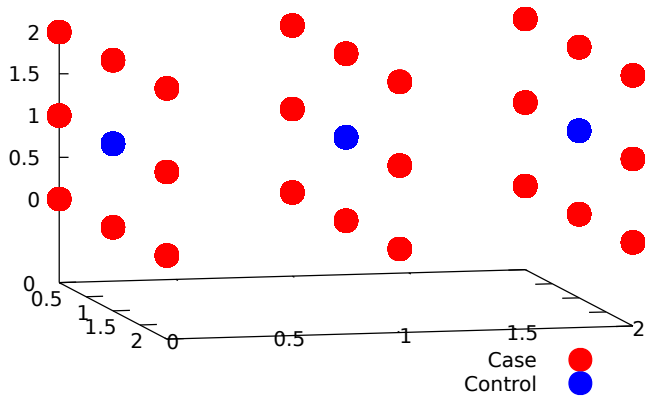

Figure 4.9: Third order interaction pattern O_P_Q.

the plot, both DCLN CE and SE methods give very similar results. Among all seven studies, the SVM with RBF kernel shows slightly better results than DCLN methods in general, but the performance between the two methods are quite close to each other, making DCLN a highly competitive classification method.

As a general trend, the performance of all methods decrease as the noise level increases, which is expected. It is easily to tell that SVM with Polynomial kernel works

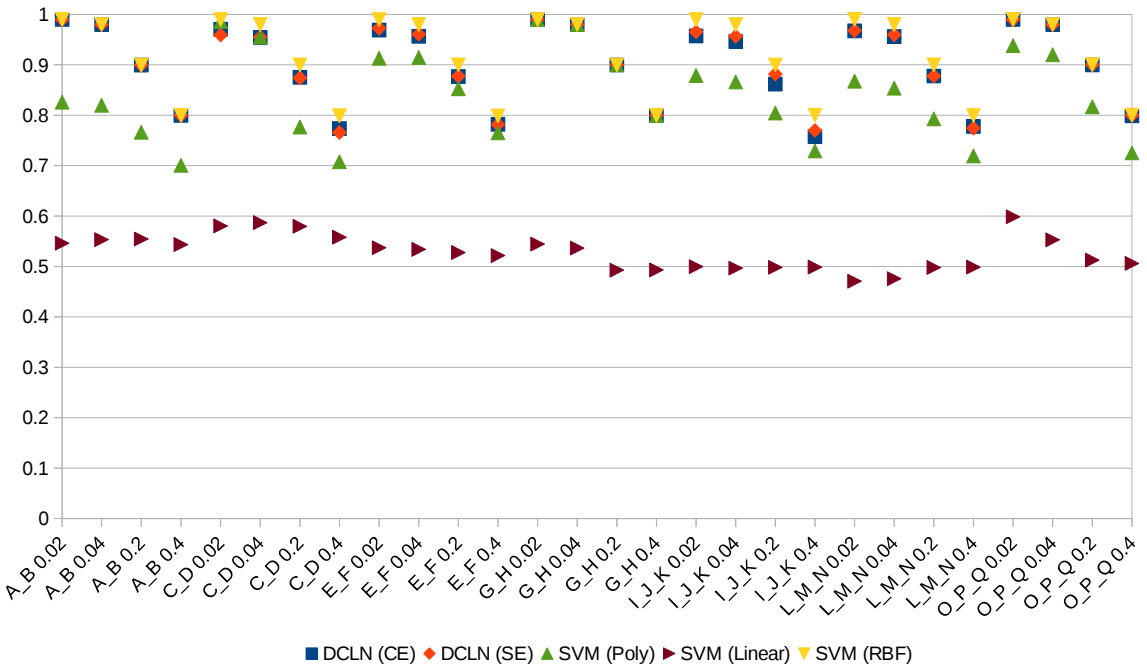

Figure 4.10: Classification performance comparision.

well for interaction pattern G_H but not for pattern A_B and three third order patterns. Additionally, it gives similar performance as SVM RBF and DCLN for pattern E_F but drops notably for C_D when the noise level exceeds 0.04 in the plot. The SVM linear kernel, on the other hand, gives the lowest performance in all scenarios, showing its inability in dealing with nonlinear studies.

In summary, the optimal SVM performance relies on the proper selection of kernel functions. It is unknown *a priori* which kernel(s) or kernel combinations are most suitable for an interaction study. For the presented simulation studies here, the RBF kernel performs better than the Polynomial kernel in general. However, when we benchmark these kernels in Chapter 5 for real GWAS tests (results are shown in Table 5.1 of Chapter 5), the situation is opposite, in which the polynomial kernel outperforms the RBF kernel in most cases. As a result, a kernel engineering is inevitable for SVM. The selection for a set of suitable kernels and the optimisation of kernel parameters add another layer of complexity to the experiment.

In contrast, DCLN doesn't rely on such selection but still works reasonably well for both simulation tests shown here and real GWAS studies shown in Chapter 5. With the

convincing classification capability, DCLN is demonstrated to learn critical information from the training data, and consequently we are interested in exporing its feature selection capability. As a first step, we would like to know, from learned weights, what features are deemed important for a binary classification.

Instead of directly stepping into the epistatic interaction topic, we begin our discussion with several computer vision experiments (each one is a binary classification problem). From those experiments, we will see soon that DCLN is able to learn complex feature dependencies; derive feature scores that are visually understandable; and effectively distinguish the signals from the noise for the binary classification. The reason of choosing computer vision applications as a starting point is because there is a ground truth of telling which are signals and which are noise by ourselves, and from a human's perspective we can easily judge whether the learned information makes sense, and hence whether DCLN works in a way similar to us to be considered more trustworthy. Once DCLN is proven to be working as expected, we apply to nonlinear interaction studies to show how epistatic interaction signals can be extracted for effective classification.

4.5 Learning High-level Features

Measuring the effect of a variable has been a common interest in statistical machine learning field. A simple example of interpreting such effect could be examining the regression coefficients of a linear regression model. For more complex systems like neural networks, such interest of assessing the features also exists by interpreting the learned weights. In Chapter 2, we have already seen that effort has been made in that direction for both shallow and deep neural-net models. Due to the complexity of the neural network architecture, the interpretation is quite challenging and usually relies on a specifically defined metric to score features. However, there is no gold stand to define such statistic to evaluate the importance of a feature, and the actual network setting could be an influential factor when proposing a novel measurement. For example, the following two measurements are proposed to evaluate the importance of a feature i by interpreting a single hidden layer network [117] and a double hidden layer network [79] respectively:

- Single hidden layer interpretation: $\sum_{k=1}^{O}\sum_{j=1}^{H}|w_{ij}w_{jk}|$ or $\sum_{k=1}^{O}|\sum_{j=1}^{H}w_{ij}w_{jk}|$, where w_{ij} represents a weight linking an input node i to a hidden node j, while w_{jk} represents a weight linking a hidden node j to an output node k.
- Double hidden layer interpretation: $\sum_{l=1}^{O}\sum_{k=1}^{H}\sum_{j=1}^{H}|w_{ij}w_{jk}w_{kl}|$, where the subscripts i, j, k and l represent the indices for the input layer, the first hidden layer, the second hidden layer and an output layer respectively. Both hidden layers share the same dimension of H.

Another method, such as [114], measures an input relevance (R_I) for a feature i using the weights of the input features for a back propagation network. To our understanding, the form of such metric for a network architecture (e.g. the simplest one) can be defined by the following equation:

$$R\_I_i = \frac{\sum_{j=1}^{C} w_{ij}^2}{\sum_{i=1}^{I}\sum_{j=1}^{C} w_{ij}^2}, \tag{4.24}$$

where I and C represent the input layer dimension and the connecting layer dimension respectively.

As for DCLN, the simplicity of its physical architecture greatly reduces the complexity of proposing a suitable metric since we only need to consider a single hidden layer which is also the classification decision layer for DCLN. During the training, the two centroids at the hidden layer move away from each other, aiming at an orthogonal relation between the two such that the structural differences are maximally learned in perfect situation. In such a case, each hidden neuron j serves as a separator, trying to enlarge the discrepancy between the two classes. Each input node i contributes to a hidden neuron j by its weight w_{ij} and an average contribution to the classification decision layer (comprised of hidden neurons) can be approximately characterised as $(\sum_{j=1}^{\beta} w_{ij})/\beta$, where β is the number of total hidden neurons. By removing the denominator constant β, the importance of a feature i can be simply defined as $\Omega_i = \sum_{j=1}^{\beta} w_{ij}$.

As can be seen that both Ω and R_I are applicable to score features for DCLN since they both measure the feature contribution by using the weights of the input nodes. Apparently, other metric candidates are potentially applicable as well, but we are not going to enumerate them here. For simplicity, we use Ω and R_I in the following computer

vision applications to illustrate the scored features.

Firstly, we obtained a human face database `https://courses.media.mit.edu/2004fall/mas622j/04.projects/faces/` from MIT media lab, containing pictures of different categories. We focus on the following binary classification problems: Smiling/Serious, Male/Female, Child/Senior, Caucasian/African, Asian/African and Caucasian/Asian. Each picture (instance) is in $128 \times 128 = 16384$ dimensions and each pixel (feature) is in gray-scale, ranging from 0 to 255 for DCLN. Several pictures sampled from the database are illustrated in Figure 4.11.

Figure 4.11: Human face examples (MIT media lab).

We train Smiling/Serious and Male/Female with $\beta = 10$, $\eta = 10$, $C = 1000$ and $epochs = 100$; train Asian/African and Child/Senior with $\beta = 20$, $\eta = 10$, $C = 10000$ and $epochs = 300$; and train Caucasian/African and Caucasian/Asian with $\beta = 30$, $\eta = 10$, $C = 10000$ and $epochs = 100$. Total 200 repeated trials are performed to contain the randomness of the neural-net model. After each trial r of training, an Ω_i^r and a $R\_I_i^r$ are calculated for each feature i respectively. Finally, the average $\overline{\Omega_i}$ and $\overline{R\_I_i}$ over all 200 trials are calculated for the feature i. For simplicity, we use Ω and R_I to denote the score averages for all features, with their values visualised in Figure 4.12.

As can be seen from the figure, the odd and even columns correspond to R_I and Ω results respectively. At a glance, key human facial areas are highlighted for each study, and R_I and Ω identify highly similar regions. For Smiling/Serious study, the mouth and eye areas are clearly distinguished from the background noise, while other facial details are minimised since they are irrelevant to the classification. The identification of these two areas agrees with the human's understanding because the muscles around the mouth

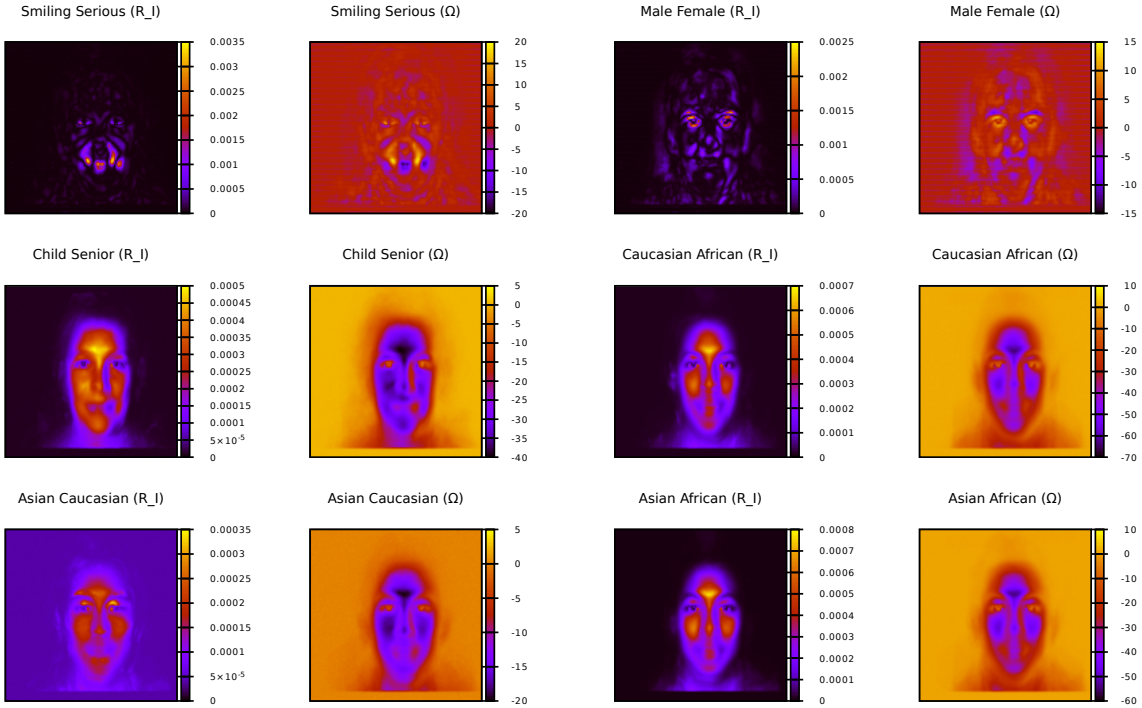

Figure 4.12: Visulisation of feature scores for binary classifications of facial classes. The Ω plots are revised from the original pictures in our paper (©2015 IEEE).

are key information for us to tell whether a presented human face is smiling or not. Also as a common sense, we can judge whether a person is happy or unhappy from the eye area. For Male/Female, areas such as beard, eyebrow, nose and eye, are detected. In our daily life, we also distinguish the gender by checking these areas. Generally speaking, for example, a female may not have a heavy beard than a male in most cases, while we may see a male with a thicker eyebrow more frequently than a female. Male and female can also differ in the nose size and eye shape. Lastly, the hair region emerged slightly from the plot, and this makes sense to us since we often judge the gender by hair (e.g. a long hair is more commonly appeared in the female group). Interestingly, such region vanished for all other studies. This is understandable because for all other studies the condition of the hair is totally irrelevant to those classifications. Next, for Child/Senior study, we see a large portion of facial area is highlighted. A possible explanation is that the overall facial muscles, skin condition and the shape of the head may all change over time, and correspondingly a large area of activated features reflects this complex. Finally,

for the remaining three ethnicity studies, the cheeks, forehead, chin and a slight nose region are learned to distinguish the race groups.

In summary, the high-level, class-specific and human-understandable features are learned by DCLN at the input layer, while the background noise that has no contribution to the classification is smoothly coloured. The learned features tell the discriminative information at a concept level which agrees with the human's cognition, and either R_I or Ω can be used to present such information. Although both metrics detect useful features, R_I allows a lower bound of zero, representing the least important features. This brings some convenience for extracting predictive signals from the noise, and we use it in later sections and chapters to demonstrate how such signals can be collected.

Next, we perform the experiment for another computer vision application using the popular MNIST handwritten digit database [70]. This database contains handwritten digit images in 10 categories $\{0,\ldots,9\}$. Each image is in $28 \times 28 = 784$ dimensions. Several randomly selected sample images are illustrated in Figure 4.13.

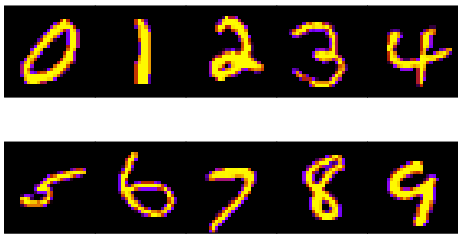

Figure 4.13: The MNIST handwritten digit examples.

Because there are 10 categories, the total number of binary classification studies is $10 \times (10-1)/2 = 45$. For each study, we run DCLN with $\beta = 10$, $\eta = 0.1$, $C = 10000$ and $epochs = 200$. Again, total 200 trials are performed to calculate the average scores, with the results shown in Figure 4.14.

As can be seen from the plots, the signal regions are generally characterised in the shape of *layered union* of the two class concepts. Take the study of discriminating digit 0 and digit 1 for instance, the region describing the shape of digit 1 floats on top of the shape of digit 0. Therefore, the concepts of both classes are well learned and represented at different activation levels. Apparently, the learned class-specific features are useful for a binary classification and are agree with our common sense. Finally, we observe

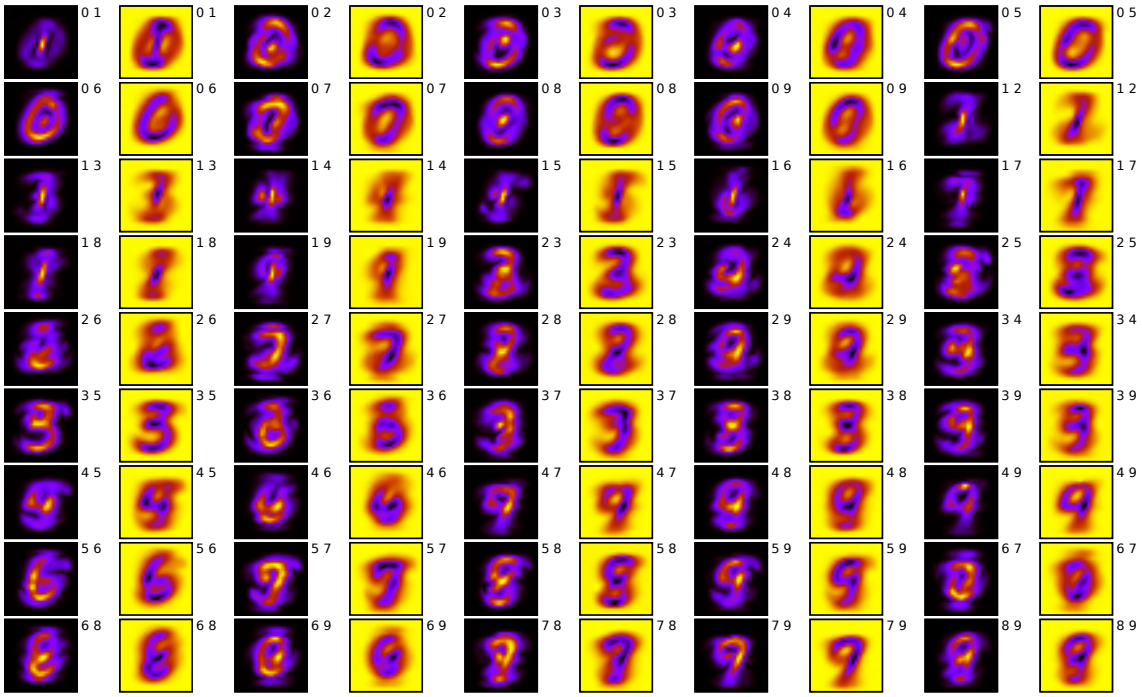

Figure 4.14: A complete set of scored feature plots for all pairs of digits.
Several Ω plots are revised from the original pictures in our paper (©2015 IEEE).

similar patterns from other plots in the figure as well, with the background noise clearly distinguished from the signals in each plot.

In summary, DCLN is shown to learn concept-level features that make sense to humans. Because the model is demonstrated to be working in a reasonable way by a number of applications, we have an increased confidence in its ability of generating meaningful features that are more trustworthy for a binary classification problem.

4.6 Extracting Predictive Signals

In the previous section, DCLN is shown to be able to assign reasonable scores to input features such that the signals and the noise are intuitively distinguishable from a human's perspective. However, for other applications such as interaction feature detection, there is a lack of a ground truth of telling whether a scored feature is a signal or not. Therefore, in addition to a set of properly scored features, we also need an automatic way of selecting only the discriminative ones based on those scores. More precisely, a routine is

required to determine a threshold from the score spectrum such that the signals and the noise can be well separated according to that threshold. For instance, we can manually select a threshold, say 0.0001, from the score spectrum of the R_I plot of the previous Smiling/Serious study, and allocate all features with scores greater than this threshold as the signals, leaving all features with scores equal to or smaller than the threshold as the noise. In this section we show how such threshold can be determined.

Given a set of scored features, the systematic way of distinguishing the signals and the noise should be an important topic in the signal processing research field, but this is out of our knowledge and expertise at this time, and we only attempt at a simplest way of addressing the problem here. Same as the principle of many feature selection methods, the objective is simply generating a set of informative features such that no noise features are included, and at the same time creating a set of noise features without the inclusion of any signals. In other words, the false positives in the signal set and the false negatives in the noise set should be minimised.

Assuming a pair of signal and noise sets are created at a given time, an evaluation criteria can be inspecting their respective performance on the validation data to see whether the signal set possesses the most discriminative power and whether the noise set gives very limited prediction capacity as expected. However, simply evaluating the performance in such a way is insufficient to infer a desired pair of signal and noise sets because a resulting signal set may contain a large portion of false positives, while it still gives a quality performance score as a whole. Instead, it is more appropriate to start with a minimal noise set, expanding its size by gradually adding least important features until the false negatives start to increase.

Following this logic, we can firstly train DCLN on the training data for a total number of R trials and record the training solution $\{\mathbf{w}^r, \mathbf{v}^r_{\mathcal{A}}, \mathbf{v}^r_{\mathcal{B}}\}$ and the feature scores R_I^r for each trial r. After that, we calculate an average score $\overline{R\_I_i}$ for each feature i, and then sort them from the smallest to the largest. We gradually take each sorted score, from the smallest to the second largest one, as the threshold θ of the current round, and assign zeros to all features with scores smaller than or equal to θ, leaving the rest features as they are. After that, the validation samples are mapped from the input layer to the hidden

layer for each trial r via $\mathbf{w}^r$ and compared to $\mathbf{v}^r_{\mathcal{A}}$ and $\mathbf{v}^r_{\mathcal{B}}$ to perform the classification and record the performance number for the signal set as $Pred(P)$. By averaging over R trials, we end up with an average performance $\overline{Pred(P)}$. In contrary, we can assign zeros to all features of the validation data with scores greater than θ and leave the remaining ones as they are. By performing the prediction mapping as before, we end up with an average performance $\overline{Pred(N)}$ for the noise set accordingly.

Obviously, the signal set starts with almost the complete set of features except for the first one being labeled as zero and regarded as a noise in the first round. As the threshold gradually increases for each round, more and more least important features are replaced with zeros and are exempted from the validation procedure until the final step, at which only the feature with largest score survived and actually contributed to the validation for the signal set, while all other features are marked as zeros in the end. In contrast, the noise set begins with a least important feature (all other features are replaced with zeros), and as the validation proceeds, more and more features are unlocked and add values to the validation procedure until the very end, where it contains all features except for the last one. As a result, the $\overline{Pred(P)}$ for the first threshold should give a quality validation result, and the $\overline{Pred(N)}$ shows very limited predictive power on the other hand. Contrarily, for the last threshold, the $\overline{Pred(P)}$ exhibits restricted performance since most features are not used by the validation procedure, while the $\overline{Pred(N)}$ gives a high quality result since it contains most features at last.

To demonstrate how the $\overline{Pred(P)}$ and $\overline{Pred(N)}$ change over the threshold, we apply the performance validation procedure to a binary classification task using the MNIST dataset to distinguish the digit 0 and digit 1 as our first case study. The total number of scored features are 784 for this study, and hence there are total 783 thresholds. For each threshold θ, we end up with a $\overline{Pred(P)}$ and $\overline{Pred(N)}$ respectively, measured in balanced accuracy. We plot the $\overline{Pred(P)}$, $\overline{Pred(N)}$ and $\overline{Pred(P)} - \overline{Pred(N)}$ for every θ in an ascending order (from the left to the right) in Figure 4.15.

As can be easily seen that, as the θ increased to around to the 500^{th} threshold, the balanced accuracy of the noise set increased above 0.5 because it starts receiving some predictive signals that are dropped by the signal set. At the same time, the balanced ac-

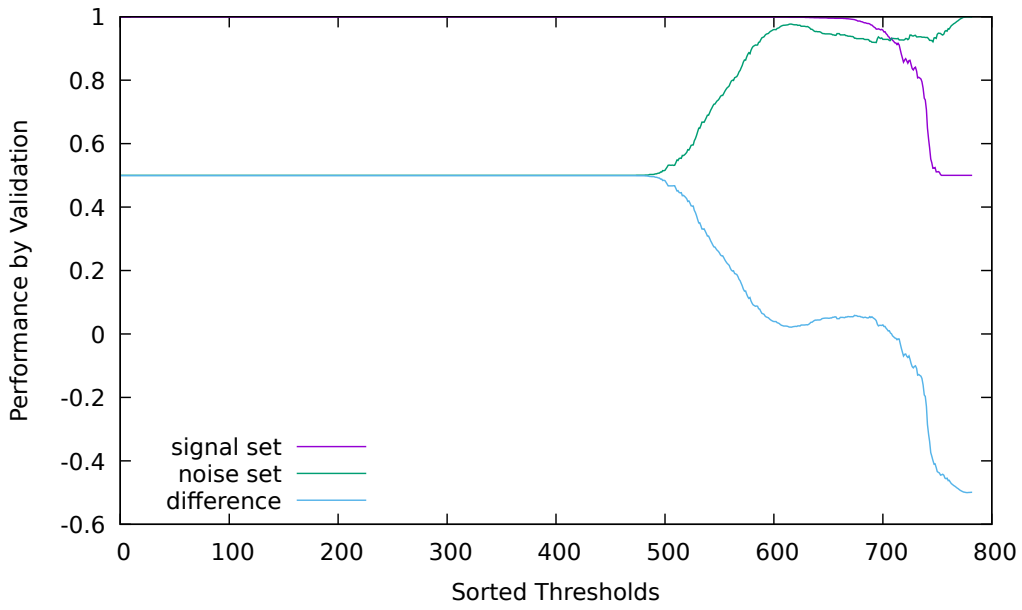

Figure 4.15: The performance validation by predition mapping over a gradually increased threshold θ. Case study: MNIST digit 0 and 1.

curacy of the signal set remains at the highest level since it still contains sufficient number of predictors. However, the situation starts to change roughly after the 600^{th} threshold, where the performance of the signal set begins to drop since some import features are lost at that time. Finally, the predictive power of the signal set vanished to around 0.5, while the noise set reached to the top level performance.

Next, we apply the same procedure to the previous facial image classifications. For those studies, the total number of features are 16384 and correspondingly we have total 16383 thresholds. This time we focus on three representative studies: Smiling/Serious, Male/Female and Child/Senior. Again, the evaluation results for the two sets over different threshold θ are plotted in Figure 4.16, 4.17 and 4.18 respectively.

From those plots, we see similar phenomena like what we observed for the MNIST study. In the beginning, the $\overline{Pred(P)}$ and $\overline{Pred(N)}$ start with a high and low performance respectively, and in the end their performance switch to the other way round. Given three lines shown in each plot, we know where the signal set starts to lose critical features and where the noise begins to take predictive signals. As a result, the objective is to find a threshold such that the false positives and false negatives are both minimised.

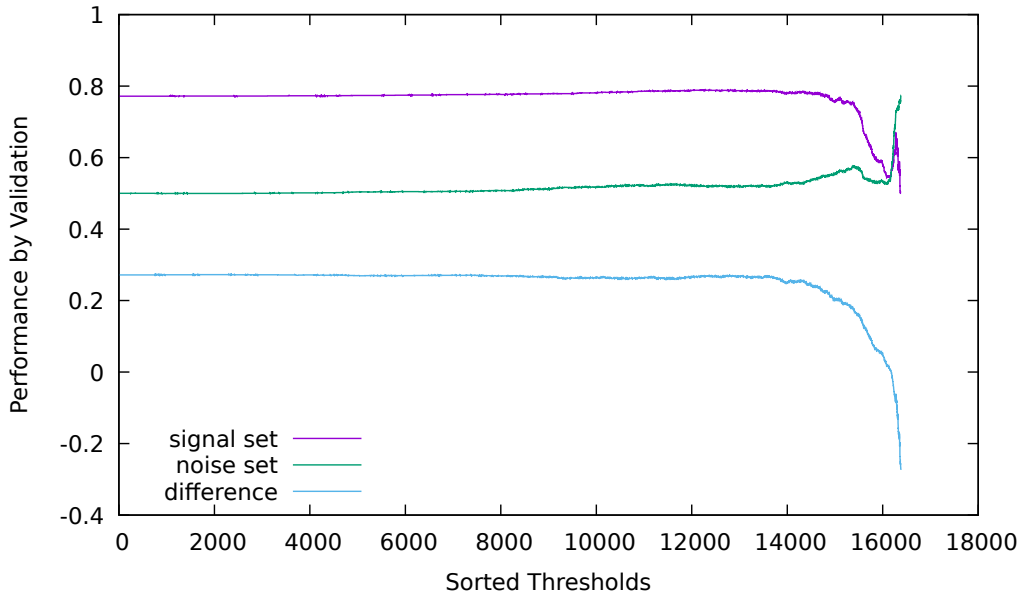

Figure 4.16: The performance validation by predition mapping over a gradually increased threshold θ. Case study: facial classes Smiling/Serious.

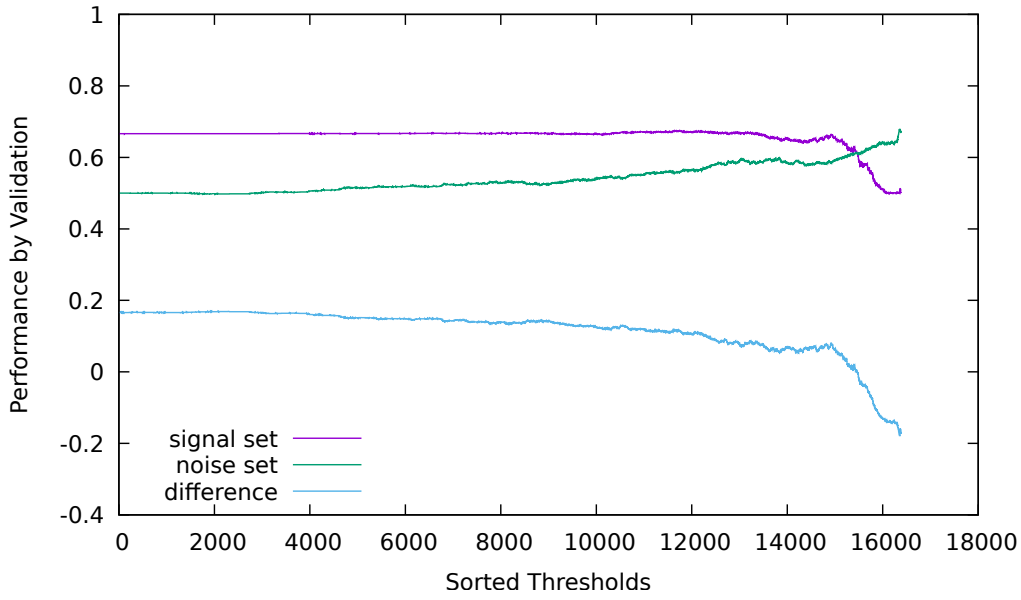

Figure 4.17: The performance validation by predition mapping over a gradually increased threshold θ. Case study: facial classes Male/Female.

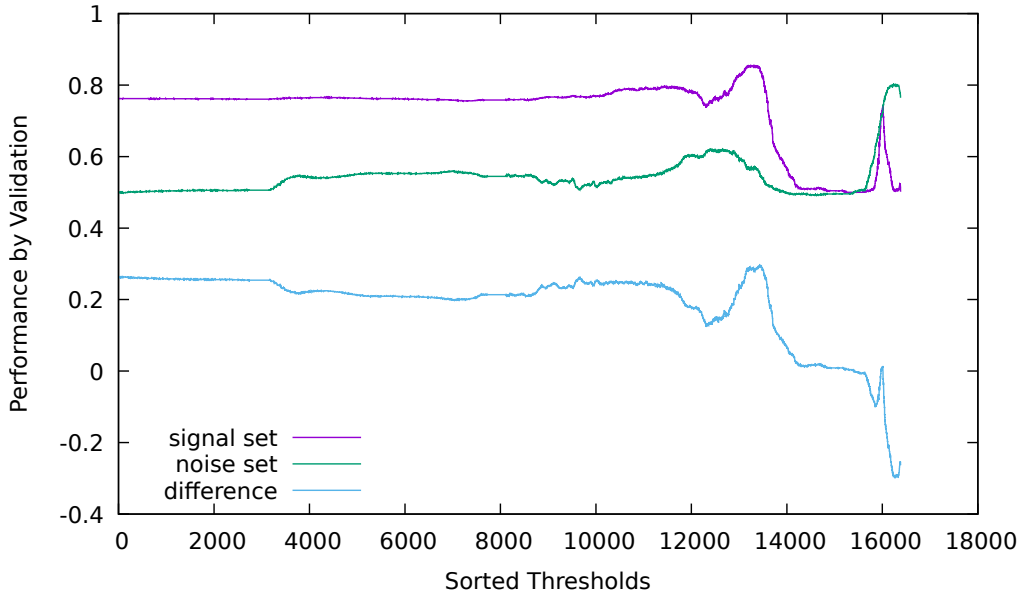

Figure 4.18: The performance validation by predition mapping over a gradually increased threshold θ. Case study: facial classes Child/Senior.

As can be seen from the plots of all four studies, it is tricky to find an ideal separating threshold and apparently various strategies are applicable. Take the Figure 4.15 for instance, we may manually select a threshold around 500 since that is the point where the green line (noise set) takes off, while the purple line (signal set) remains at the highest level. In other words, the false negatives in the noise set start to increase around that point. At the same time, the blue line (discrepancy between the two) begins to drop notably. For Figure 4.16, both purple and green lines remain steady until they reached to around the 14000^{th} threshold, where the signal set starts giving informative features to the noise set. As a result, a drop of the blue line is observed after that point. Next, for Figure 4.17, the green line slowly climbs up after $\sim 4000^{th}$ threshold. Therefore, we may wish to pick a threshold around that place to stop the inclusion of false negatives into the noise set. Lastly, for Figure 4.18, the peak of the blue line seems to appear around $13000 \sim 14000$. After that, it decreases dramatically.

To select a threshold, it can be simply as taking the one that is associated with the max $\overline{Pred(P)} - \overline{Pred(N)}$, where the peak of the blue line occurs. However, this may not be the best strategy. For instance, in Figure 4.15, the discrepancy between the signal and the noise sets remains almost a straight line until it closes to $\sim$ 500. If we only

consider the largest discrepancy value, then presumably its corresponding θ falls into somewhere between $0 \sim 400$. Assuming the 200^{th} threshold is found to be the one with largest discrepancy between the two sets, then the signal set at that place still contains a large volume of false positives, until all those false positives are cleaned up at around the 500^{th} threshold. Consequently, it is better to address this issue by allowing some level of flexibility. Instead of directly using the $\max \overline{Pred(P)} - \overline{Pred(N)}$, a simplest patch could be considering a lower bound $(1-\lambda)\max \overline{Pred(P)} - \overline{Pred(N)}$, where $\lambda \in (0,1)$, and select the largest possible θ^* whose $\overline{Pred(P)} - \overline{Pred(N)}$ is no less than the lower bound. By looking at the curves, this means that the false positives in the signal set are minimised at the cost of allowing certain degrees of degradation of the blue curve. After solution θ^* is determined, the signals and the noise can be separated.

To demonstrate the outcomes of the lower bound discrepancy strategy, we set $\lambda = 0.01$ for MNIST digit $0/1$ study and set $\lambda = 0.1$ for the human facial studies. It turns out to be total 491, 14491, 4862 and 13537 features are identified as the noise for the studies of MNIST digit $0/1$, Smiling/Serious, Male/Female and Child/Senior respectively. These numbers closely matched to the ones that are determined empirically by us in the previous discussion. To visualise the signals and the noise that are separated by different θ^* (derived from different levels of λ), we plot the recognised signal features with their original feature scores and plot the recognised noise features with zeros in Figure 4.19.

In the figure, the first row represents a reference to the original R_I score before separation, and the presented information is easily understood by a human. The second row corresponds to the desired separation results generated by the lower bound discrepancy strategy. Thus, the validation procedure itself is able to tell which are signals and which are noise. As can be seen, the predictive signals that are critical to the binary classifications are well identified and separated from the noise. The last row represents undesired separation results, generated by a threshold which is derived from an unreasonably large $\lambda = 0.9$. In such a case, total 692, 16064, 15413 and 14167 features are identified as the noise respectively for MNIST digit $0/1$, Smiling/Serious, Male/Female and Child/Senior. If we match these four numbers back to the corresponding positions in Figure 4.15, 4.16, 4.17 and 4.18 respectively, then we can easily see that: 1) a large

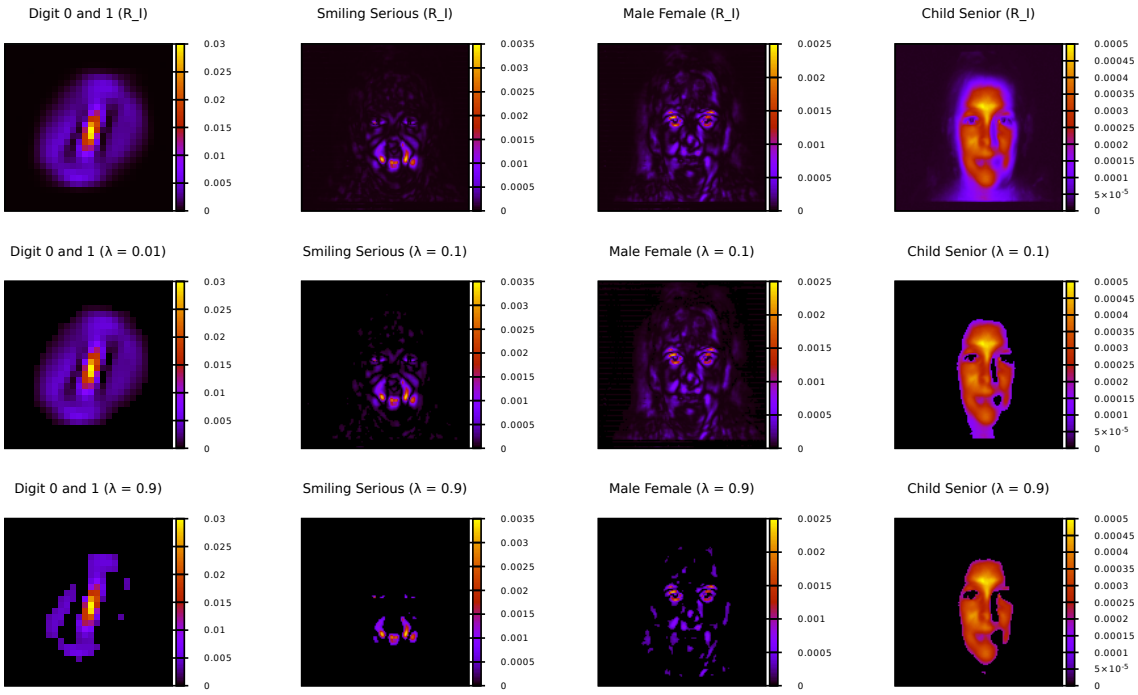

Figure 4.19: Predictive signal extraction at different levels of λ.

number of false negatives are included into the noise set for digit 0/1 study; 2) the signal set loses critical informative features for Smiling/Serious study; 3) the signal set keeps dropping predictors to the noise set and the two lines are close to the cross-over point for Male/Female study; and 4) the discrepancy between the signal and noise sets is greatly reduced for Child/Senior study. If we link these observations to the last row of Figure 4.19, then we see that some important regions for effective binary classification are missed. For instance, the shape of digit 0 is no longer complete; the critical mouth and eye regions shrank for Smiling/Serious study; and the hair region together with a portion of other facial details disappeared for Male/Female study. Therefore, the λ should not be set with overly large values.

From these tests, it is clear that DCLN scores features in a reasonable way and end up with concept-level feature understanding which is usually considered as an asset of deep architectures. Additionally, it allows a simple validation procedure to actually separate the discriminative features from the noise based on those feature scores with minimum computational cost. During the validation, only previously generated training solutions

are used for mapping instances to the hidden layer to derive the prediction performance and no expensive re-training of the model is required. In the next section, we show how this procedure is applied to learn nonlinear discriminative interactions.

4.7 Learning Interaction Features

From previous discussions, the proposed DCLN method is shown to perform effective binary classification; score features in a reasonable way; and extract desired predictive signals by prediction mapping. Via a series of demonstrations, we developed an increased confidence in its capacity of producing meaningful features. In this section, we apply the method to learn predictive nonlinear interactions. We wish to know whether it still score features in a reasonable way for epistatic interaction studies and whether the prediction mapping is still applicable to extracting predictive signals for this type of analysis. We demonstrate its signal extraction ability for nonlinear interactions by performing simulation tests here and we leave real GWAS analyses to Chapter 5.

Again, we focus on previous four nonlinear studies shown in Table 2.4, and visualise their raw data in Figure 4.20. In the figure, each row represents an instance and each column represents a SNP. There are 4 nonlinear patterns in total 8 SNPs: 'A', 'B', 'C', 'D', 'E', 'F', 'G' and 'H'. The three possible genotype values 0, 1 and 2 are coloured in 'white', 'blue' and 'black' respectively in the datasets. Each group, either Case or Control, contains total 1000 instances, and each genotype combination receives their respective samples based on the sample allocation specified in Table 2.4.

The objective of the first experiment is to check whether DCLN scores features in an expected way. In short, we wish to visualise the feature scores for a pair of SNPs and see how their scores change if the synergistic effect between the two SNPs is broken. Assuming we connect all 8 SNPs together to form a single dataset with 8-dimension input features 'A-B-C-D-E-F-G-H', if we randomly shuffle the sample order in SNP 'A', then its synergistic effect with 'B' is broken. In such a case, we expect to see that both SNPs 'A' and 'B' are scored as noise features because they are not effective predictors independently on their own. Therefore, if we feed this 8-dimension input data to DCLN for feature

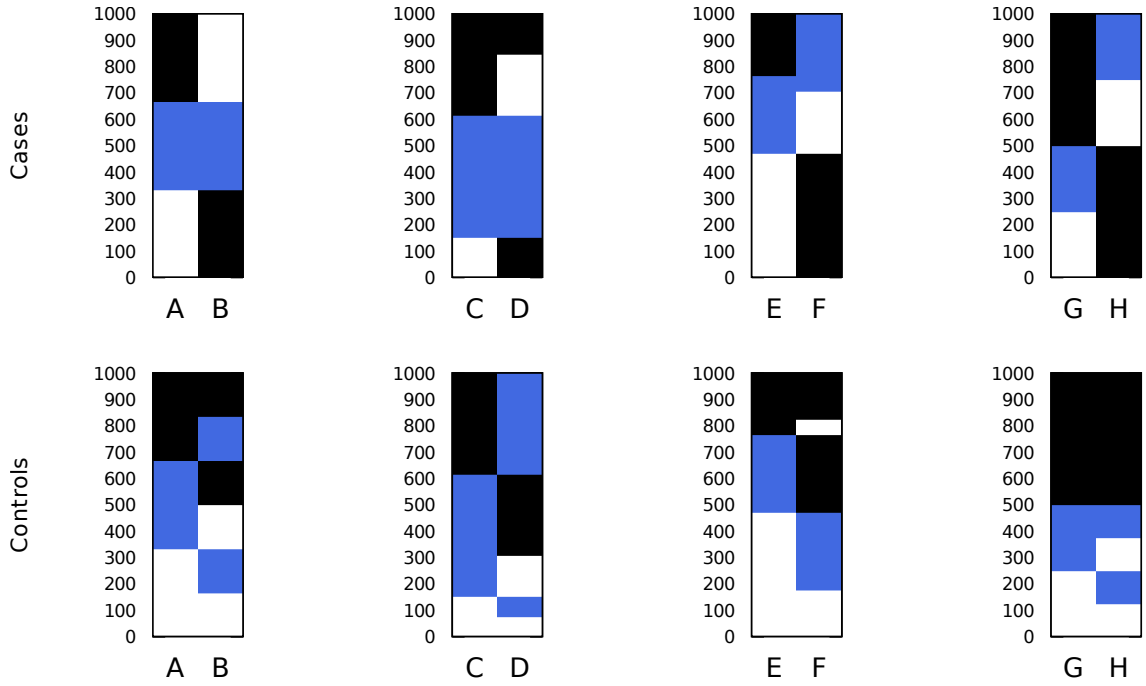

Figure 4.20: Raw data of four nonlinear interaction patterns given by Table 2.4. Each row represents an instance and each group contains total 1000 instances. The genotypes 0, 1 and 2 are represented by 'white', 'blue' and 'black' colours respectively.

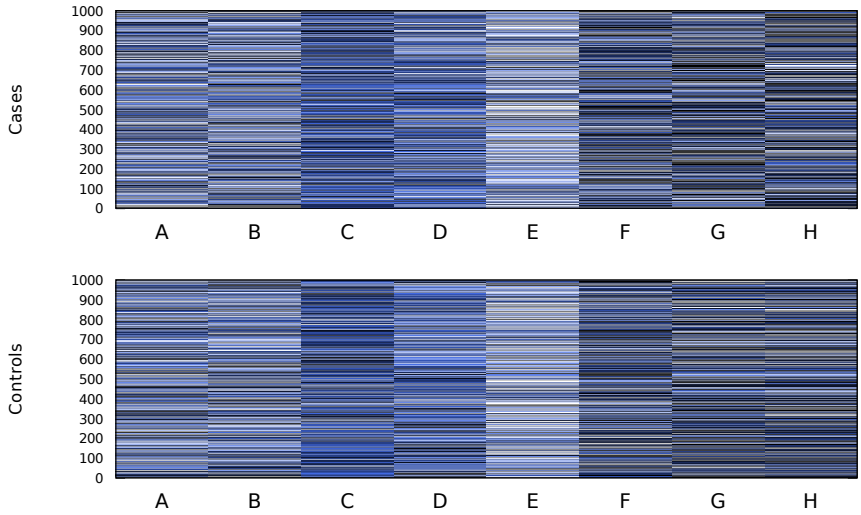

Figure 4.21: In each group, either Case or Control, the samples are shuffled for each SNP-pair independently and then all shuffled SNP-pairs are combined to form a single dataset at a given trial r.

scoring, then the feature scores (measured in R_I) of 'A' and 'B' should be very close to zero (least important features), while the remaining 6 feature scores are expected to be visually deviated from zero in theory. However, we can not feed this 8-dimension input data directly into the neural net using its raw form (illustrated in Figure 4.20) because we need to first break the possible dependencies between different SNP-pairs. Imagine the samples in SNP 'A' are shuffled, however in such a case, SNP 'B' may still be regarded as an effective predictor because it may possibly pair with one of the remaining 6 SNPs and form a synergistic effect there as an epistatic interaction. To diminish the possible inter-SNP-pairs dependencies, we shuffle the samples independently for each of the 4 SNP-pairs and then connect all SNPs together as a signal dataset at a trial of the experiment. By doing so, the synergistic effect within a pair of SNPs is kept, while such effect to other possible SNPs are minimised. A shuffled version of the dataset is illustrated in Figure 4.21 for a given experiment trial r.

Next, at each trial r, we generate 14 datasets from the originally shuffled dataset with each one corresponds to a SNP-masking scheme, which means certain SNPs in the dataset are replaced with random number $\{0,1,2\}$ to ensure those SNPs become random noise and have no contributions to the classification. The 14 masking options are summarised in Table 4.1. Take the first entry in the table for instance, the masked SNP is 'A', and consequently the expected noise SNPs are 'A' and 'B'. Therefore, we have 6 expected SNPs in the signal set with a maximum possible false positives (FPs) of 2, and on the other hand, there are total 2 expected SNPs in the noise set with a maximum possible false negatives (FNs) of 6. At this stage, we wish to perform experiment for these 14 scenarios to check whether the generated feature scores are visually distinguishable between the expected signals and the expected noise. If the feature scores indeed make sense, then we proceed to the next experiment to check whether the validation procedure can correctly separate the signals from the noise. At the moment, we check the feature scores first.

Total 4000 trials are performed. For each trial, half of the samples are randomly selected from the dataset for model training, and DCLN is trained with $\beta = 100$, $\eta = 100$, $C = 1$ and $epochs = 400$. The recorded feature scores R_I for all 4000 trials are illustrated in Figure 4.22 for each scenario. Take the first plot from the figure for instance, each row

Table 4.1: Total 14 different SNP-masking scenarios are listed. Once specific SNPs are masked, their genotype values for all instances in the dataset are replaced with random numbers $\in \{0,1,2\}$ to ensure those SNPs are random noise.

Masked SNPs	Expected signals (blue) and noise (black)	Max FPs (signal set)	Max FNs (noise set)
A	A-B-C-D-E-F-G-H	2	6
C	A-B-C-D-E-F-G-H	2	6
E	A-B-C-D-E-F-G-H	2	6
G	A-B-C-D-E-F-G-H	2	6
AC	A-B-C-D-E-F-G-H	4	4
AE	A-B-C-D-E-F-G-H	4	4
AG	A-B-C-D-E-F-G-H	4	4
CE	A-B-C-D-E-F-G-H	4	4
CG	A-B-C-D-E-F-G-H	4	4
EG	A-B-C-D-E-F-G-H	4	4
ACE	A-B-C-D-E-F-G-H	6	2
ACG	A-B-C-D-E-F-G-H	6	2
AEG	A-B-C-D-E-F-G-H	6	2
CEG	A-B-C-D-E-F-G-H	6	2

corresponds to one of the total 4000 trials and each column represents a SNP. The feature scores R_I are visualised in the plot with its lower bound 0 shown for the least important features. As can be seen that the feature scores of 'A' and 'B' are both very close to 0, while other features deviate from 0 at various levels. Therefore, the features are scored in a reasonable way by DCLN for this scenario. We can also see from other plots that the noise features are all properly scored (pushed toward 0), and DCLN is indeed aware of the differences between the nonlinear interaction signals and the noise in all scenarios. From the presented results, DCLN is proven to be flexible to learn different nonlinear patterns in various combinations. At the same time, the generated feature scores are smooth across 4000 trials, showing the stability of proposed method.

Next, we proceed to the signal extraction experiment, showing the signal-noise separation results. This time, we increase the difficulty of the experiment by introducing

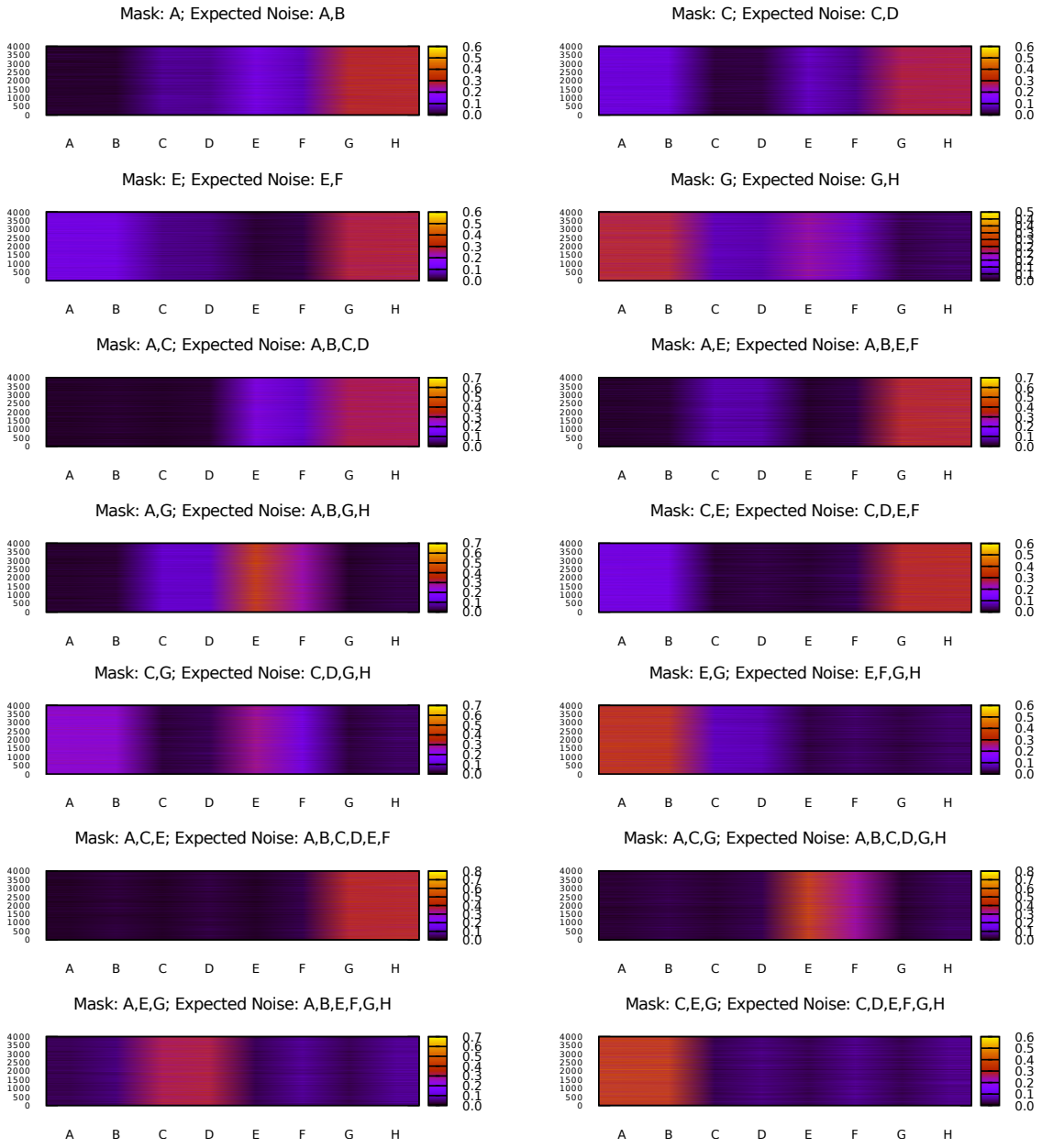

Figure 4.22: Total 4000 feature scoring results (R_I) for each SNP-masking scenario.

random noise into the datasets to see whether DCLN can correctly score and extract the expected signals under different random noise conditions. We are interested in seeing how many SNPs are correctly identified in the expected signal and noise sets respectively for each scenario under an increased level of injected random noise.

In response, for each of the 14 SNP-masking datasets, we create 4 noise-enabled ver-

sions of data by randomly selecting a portion of samples from the dataset and replace them with random numbers $\{0,1,2\}$. Again, we use the noise levels at 0.02, 0.04, 0.2 and 0.4 of the original sample size. As a result, we have total 5 dataset versions for each SNP-masking scenario, with their noise levels at 0.0, 0.02, 0.04, 0.2 and 0.4 respectively. To perform signal-noise separation, we still use the previous prediction mapping-based validation procedure by calculating an average feature score for each feature; sort the average scores from the smallest to the largest; perform prediction mapping for each θ and record $Pred(P)$ and $Pred(N)$ at every step; determine λ (e.g. $\lambda = 0.1$ is used here) and find the largest possible solution θ^* with its $\overline{Pred(P)} - \overline{Pred(N)} \geq (1-\lambda) \max \overline{Pred(P)} - \overline{Pred(N)}$; and finally allocate SNPs to the signal set by checking whether their average feature scores are greater than θ^*.

There are total 14 scenarios $\times$ 5 dataset versions $=$ 70 studies for this experiment. For each of them, we check how well the signals are correctly distinguished from the noise by calculating the number false positives (FPs) and false negatives (FNs) appeared in the expected signal and noise sets respectively. Finally, the compete signal-noise separation results for the 70 studies are presented in Table 4.2. From the table, take the 'AE-masking' study for instance, the expected signal set contains 'C,D,G,H' and the expected noise set contains 'A,B,E,F'. If we check the actually detected SNPs for the dataset with random noise level 0.4, then the observed signal set from the experiment is 'A,C,D,E,G,H' and the detected noise set is 'B,F', hence two false positives 'A,E' appeared in the signal set.

As can be seen, there is no false negative appeared in the table and only several places contain false positives. Generally speaking, the signals and the noise are well separated by the validation procedure, and the proposed method is shown to be flexible to realise different types of nonlinear interaction patterns and their combinations.

4.8 Summary

In this chapter, we have presented a shallow neural network, enabling both the binary classification and the feature selection at the same time. The selection of discriminative interaction features in GWAS requires a method to be flexible enough to identify var-

Table 4.2: The signal-noise separation results for 14 SNP-masking scenarios under 5 different random noise levels $\{0.0, 0.02, 0.04, 0.2, 0.4\}$. The detected FPs and FNs out of their respective maximum possible values are shown in the table.

Masked SNPs	Errors	0.0	0.02	0.04	0.2	0.4
A	FPs	0/2	0/2	0/2	0/2	0/2
	FNs	0/6	0/6	0/6	0/6	0/6
C	FPs	0/2	0/2	0/2	0/2	0/2
	FNs	0/6	0/6	0/6	0/6	0/6
E	FPs	0/2	0/2	0/2	0/2	0/2
	FNs	0/6	0/6	0/6	0/6	0/6
G	FPs	0/2	0/2	0/2	0/2	0/2
	FNs	0/6	0/6	0/6	0/6	0/6
AC	FPs	0/4	0/4	0/4	0/4	0/4
	FNs	0/4	0/4	0/4	0/4	0/4
AE	FPs	0/4	0/4	0/4	0/4	2/4
	FNs	0/4	0/4	0/4	0/4	0/4
AG	FPs	0/4	0/4	0/4	0/4	0/4
	FNs	0/4	0/4	0/4	0/4	0/4
CE	FPs	0/4	0/4	0/4	0/4	0/4
	FNs	0/4	0/4	0/4	0/4	0/4
CG	FPs	0/4	0/4	0/4	0/4	0/4
	FNs	0/4	0/4	0/4	0/4	0/4
EG	FPs	0/4	0/4	0/4	2/4	2/4
	FNs	0/4	0/4	0/4	0/4	0/4
ACE	FPs	0/6	0/6	0/6	0/6	0/6
	FNs	0/2	0/2	0/2	0/2	0/2
ACG	FPs	0/6	0/6	0/6	0/6	0/6
	FNs	0/2	0/2	0/2	0/2	0/2
AEG	FPs	0/6	0/6	0/6	3/6	4/6
	FNs	0/2	0/2	0/2	0/2	0/2
CEG	FPs	0/6	0/6	0/6	0/6	0/6
	FNs	0/2	0/2	0/2	0/2	0/2

ious nonlinear patterns under different conditions. Moreover, such method should be at least interpretable to some extent and being stable in feature ranking. Via a number of simulation tests, we have demonstrated that the proposed method can score features in a reasonable manner with its decision close to a human's choice. The distinguished signal and noise sets generally contain their expected features with false positives and false negatives kept at a very low level. Due to its demonstrated feature selection ability and the simplicity of its shallow architecture, we apply this method in Chapter 5 to comprehensively learn epistatic interaction features for real GWAS datasets.

Chapter 5

Learning Epistatic Interaction Features in GWAS

IN this chapter, the two-stage filtering framework is introduced, namely a protocol that combines previously proposed two techniques to learn interaction features from GWAS data. The associated technical challenges are discussed first, followed by a formal introduction to the framework. The predictive power and the learned discriminative interactions are illustrated in the end. Throughout this chapter, a breast cancer study is used to demonstrate the cross-cohort feature learning and prediction. The framework is shown to be able to learn and distinguish robust and discriminative interaction signals. By utilising fast GPU screening method and efficient shallow neural network, the high performance feature selection framework is applicable to real GWAS analyses. The following manuscript is organised partially based on this chapter[a].

[a] Qiao Wang, Aaron Harwood, Miroslaw Kapuscinski, Cheng Soon Ong and John Hopper, "Efficient learning of robust and discriminative SNP-SNP interactions for Case-Control GWAS", *Bioinformatics.* (to be submitted)

5.1 Overview

The two-stage filtering framework is formally introduced in this chapter to comprehensively identify SNP-SNP interaction features for Case-Control discrimination. The proposed framework combines the techniques introduced in Chapter 3 and 4 and encapsulates them into a re-sampling-based protocol to test the learned interactions on an independent population cohort. In the first stage processing, the re-sampling procedure is performed repeatedly on the training cohort to select robust SNP interactions. For each trial, GPU-based exhaustive tests are used as filtering methods to generate SNP-pairs that are deemed associated with the trait. Top robust SNP interactions appearing frequently

across a large number of trials are retained for further analysis.

In the second stage filtering, only robust SNPs are retained and cross-validations are performed to fine-tune the hyperparameters for the neural-net model. For the best hyperparameter setting, repeated runs of model-training on the training data enables the calculation of an average feature score for each SNP, measuring the relative contribution to the classification. Once the feature scores are calculated, a cut-off threshold can be determined based on the validation set used for performance evaluation, separating the robust SNPs into a signal set and a noise set respectively. In this chapter, the breast cancer GWAS is used for the experiment. Given limited computing resources, the complete experiment protocol is performed on independent chromosomes for signal extraction.

The objective is to systematically learn robust and discriminative SNP-SNP interaction features for Case-Control classifications. However this task is quite challenging due to a number of technical limitations. As it has been shown in early chapters, directly applying a predictive model to the whole data is computationally infeasible due to the high dimensionality of input features. On the other hand, applying a light-weight association test is practical for genome-wide scanning but it may not warrant the discovered interactions are valid predictors for a classification task. In short, the practical learning of interaction features faces the challenge of balancing the use of filtering methods and the involvement of an expensive predictive model.

As we already know that filters are efficient in feature selection but they are not connected to an underlying predictive model. This is exactly the reason why the exhaustive tests of association analyses are not suitable for discovery of discriminative interactions as there is no mechanism that can examine the predictive power of generated features during the exhaustive search. The produced SNP-pairs may contain a considerable amount of false-positives for Case-Control discrimination. To reduce such false-positives, at least two types of effort should be engaged:

- Stable feature selection: Filter methods, such as an association test, can be applied for selecting interactions that are highly associated with the trait, but a re-sampling routine should be adopted for selecting robust ones since the reported top SNP interactions from a quadratic search space by a single run may be heavily biased

to a certain set of presented training samples. By introducing variances into the training set for each trial, a portion of such false positives can be discovered and removed from a subsequent analysis.

- Performance validation of detected robust features: Since a set of robust interactions is retained, the original input dimension is greatly reduced and a performance validation procedure should be available for evaluating those robust features in terms of the prediction capacity. The ones with little predictive power are removed, while the remaining interaction features may be valuable for biological interpretation and potential applications.

In this chapter, we address these issues by showing a computationally feasible way of selecting interaction features that are highly associated with the trait; robustly against the sampling variation; and are effective for Case-Control classification. We demonstrate that the objective can be partially achieved with a reasonable amount of computational resources. We first give a formal definition of the problem; describe a number of elements required by the experiment; discuss related challenges in detail; and then introduce the processing framework. We use breast cancer GWAS as a case study to demonstrate the complete analysis procedure, with prediction performance and the learned interaction patterns shown in this chapter.

5.2 Problem Definition

The generic representation of a GWAS data can be simply denoted as $\mathcal{D} = \{\mathbf{x}_n^c, y_n^c\}_{n=1}^{N^c}$, where each instance $\mathbf{x}_n^c$ is a d dimensional vector with each feature $x_{ni}^c \in \{0,1,2\}$ representing three possible genotypes, while the superscript $c \in \{\mathcal{A}, \mathcal{B}\}$ represents two classes, either Case or Control.

Within a given training set $\mathcal{D}_T$, we wish to select a fraction of interactions, in the first stage screening, that are both in high degree of associations with the traits and robustly against the variations in training data. In response, a re-sampling procedure can be employed by randomly splitting the training samples into two parts multiple times, allocating a fraction $\mathcal{P}_f$ of samples to be a model-fitting set $\mathcal{D}_f$ and leaving the remaining

proportion $\mathcal{P}_v$ ($\mathcal{P}_v = 1 - \mathcal{P}_f$) of samples as a validation set $\mathcal{D}_v$ at each time. The generated sets of $\mathcal{D}_f$ and $\mathcal{D}_v$ are both used by the second stage filtering, but in this stage we focus on $\mathcal{D}_f$ for robust feature selection as follows.

For each trial r, out of the total number of $\mathcal{R}$ trials, an exhaustive search for pairwise interactions is performed on $\mathcal{D}_f^r$ using a pre-defined statistical test ψ to select a list of top $\top$ pairs $L_{\psi\top}^r$. Here, the list of SNP-pairs is ranked based on the raw statistic or the corresponding significance value of test ψ being used. For total $\mathcal{R}$ trials of re-sampling, we end up with total $\mathcal{R}$ $\mathcal{D}_f^r$ and $L_{\psi\top}^r$, where $r = 1, \ldots, \mathcal{R}$. From those $\mathcal{R}$ lists of pairs, we wish to further identify a number of top robust features that appear most frequently across $L_{\psi\top}^r$ as a robust list $\mathbf{L}_\psi$ of SNPs. From the derived robust list $\mathbf{L}_\psi$, we wish to further identify a fraction of SNPs $\mathbf{l}_{\psi\mathbf{P}}$ as effective predictors for Case-Control classification, leaving the rest as the noise $\mathbf{l}_{\psi\mathbf{N}}$.

The signal-noise differentiation is handled by the second stage filtering. To determine which SNPs belong to which group, either $\mathbf{l}_{\psi\mathbf{P}}$ or $\mathbf{l}_{\psi\mathbf{N}}$, we expect a method, with its optimised hyperparameters, to score those SNPs for their relative contributions to the Case-Control classification. Therefore, the first step is to apply the method on $\mathbf{L}_\psi$ using $\mathcal{R}$ cross-validations to select the optimal setting of hyperparameters. This is a common strategy of fine-tuning the hyperparameters in machine learning. Due to the large volume of computations incurred by the cross-validations, the adopted method is required to be simple and computationally efficient such that it can be tolerated by such expensive experiment protocol.

With the best hyperparameter setting determined, the method then scores each feature for its importance/contribution to the classification. Finally, we would like to determine a cut-off threshold of feature scores such that $\mathbf{l}_{\psi\mathbf{P}}$ and $\mathbf{l}_{\psi\mathbf{N}}$ are well separated. To assess the prediction performance of $\mathbf{l}_{\psi\mathbf{P}}$, we can validate on an independent test data $\mathcal{D}_t$.

Considering the complexity of the task and the high dimensionality of search space, the adopted techniques, methods and experiment protocols are required to be highly efficient, effective and robust. Due to a large volume of computations involved in the protocol and a number of experiment settings awaiting to be determined, the overall objective is mainly subject to several technical challenges, summarised as follows:

- Which statistical test ψ to be used for a particular study? Is there a rule of thumb of determining a suitable one *a priori*? Is it computationally feasible to be repeatedly used by a re-sampling procedure, given the quadratic complexity of the exhaustive search imposed by each re-sampling trial?
- Given limited computing resources, how to determine the practical number of runs $\mathcal{R}$ for cross-validations since both the exhaustive search by an association test and the predictive model training rely on total $\mathcal{R}$ model-fitting sets $\mathcal{D}_f^r$.
- How to decide the top $\top$ pairs for each trial of exhaustive search? By varying $\top$, how to balance between the significance levels of reported SNP-pairs at each trial and the cross-validation consistency of the final robust interactions? Given a generated robust list $\mathbf{L}_\psi$, how does its size impact on the predictive model training?
- Given the robust list $\mathbf{L}_\psi$, how can we create the ideal sets of $\mathbf{l}_{\psi\mathbf{P}}$ and $\mathbf{l}_{\psi\mathbf{N}}$ such that the false positives and false negatives are minimised respectively in the two sets? Different contribution scores can be assigned to the features by different scoring metrics. There is a lack of a gold standard to define such metric, and the derived feature scores rely on specific validation procedures to interpret and decide which SNPs go to which set. The applicability, performance and efficiency are main considerations when choosing a suitable validation procedure.

We give detailed explanations and discussions for each of these challenges in the following section, showing the complexity of the experiment. After that, we formally introduce the proposed framework.

5.3 Challenges

In this section, we give an overview of related challenges for data analysis.

5.3.1 Choice of an Association Test

The choice of an appropriate test ψ mainly relies on the following considerations: 1) the nature of the test ψ based on its explicit definition; 2) the computational overhead that can be tolerated by the expensive re-sampling procedure.

Firstly, a specifically defined test may favour certain types of interaction models. In Chapter 2, we have introduced several classical association tests together with their respective theoretical foundation. However it is difficult to determine which is the most suitable one for a particular study *a priori*. It is up to the intrinsic nature of the statistical test and the heterogeneity of the underlying data to decide which type of interactions to be selected by which test method. We demonstrate with a breast cancer study in this chapter that the SNPs emerged from individual chromosomes are given by different test methods ψ. Thus, traversing through a range of test candidates is necessary to locate signals for different studies.

Secondly, the stability of each association test is unknown for a given study. Without knowing the stability for a method ψ, it is impossible to ensure such method is capable of generating stable results under sampling variations, and the derived features may not be reproducible on another dataset. Hundreds, thousands or more re-sampling trials may be needed to contain the sampling variation. Taking a large volume of trials into consideration, the computational cost of a single exhaustive search performed at each trial must be minimised to ensure the overall runtime is managed at a proper scale. The complexity of the test itself and the availability of the underlying hardware and software are all critical factors to the feasibility of data processing.

For many existing tests, as we have demonstrated in Chapter 3, the most time consuming part of data processing is the contingency table calculation in GPU (e.g. heavy bitwise operations and other related instructions) for an exhaustive search. In comparison, a statistical test is a relatively light-weight procedure since it is built from the contingency table counts by a limited number of steps of light-weight mathematical operations. Therefore, methods such as χ^2, info gain, odds ratio, Pearson's correlation coefficient-based tests, etc., are all shown to be feasible for screening the interactions, at least via a GPU-based (or other hardware) acceleration technique.

However, the available computational resources may not be sufficient when there is an excessive increase in either the number of test candidates, the dimensionality of data, or the number re-sampling trials. For this reason proper balancing these factors becomes the key to robust interactions selection.

5.3.2 Practical Number of Cross-validation Trials

The cross-validations are adopted in the experiment mainly for two purposes: 1) given a sufficient number of cross-validations, one can produce a set of robust features based on the lists of interactions returned by each $\mathcal{D}_f^r$; and 2) providing a pathway of fine-tuning the hyperparameters for a classification-oriented scoring method.

In either case, adding more trials reduces the uncertainty. Without sufficient number of re-sampling trials, the reported top robust SNP-pairs may not converge to a stable set for each time the experiment is rerun. Once it is determined, the produced set of SNP features in the robust list become the input of the second stage filtering, hence they are expected to be stable for the follow-up prediction evaluation and discriminative signal extraction. On the other hand, cross-validations can also be used as a way of selecting the best hyperparameters combination. The optimal performance of a statistical model relies on the fine-tuning of a series of hyperparameters. A common way is to grid-search all possible hyperparameter combinations and test each on the validation set to determine the best performing one. Once the best setting is decided, the model can be used to score SNPs in the robust list to measure their contributions to the classification, and then the discriminative signals can be extracted accordingly based on those scores.

Although it is desired to include more trials in theory, the complexity of the experiment may restrict the total number of runs to a manageable level. Let us assume for the moment that each training set $\mathcal{D}_T$ contains 1 million SNPs. To select robust SNP-SNP interactions, total $\mathcal{R} = 1000$ cross-validations are performed within $\mathcal{D}_T$. At each run r, an exhaustive search is performed on $\mathcal{D}_f^r$ by a test method ψ. If we have 5 such candidate methods, then the total number of exhaustive search required by the whole procedure is $5 \times 1000 = 5000$, which is almost impossible to run in a traditional way because each exhaustive search requires to examine a total of $1\text{M} \times (1\text{M} - 1)/2 = 0.5$ trillion SNP-pairs. The computation time increases quadratically with the number of features and linearly with the sample size and the total number of runs $\mathcal{R}$. Therefore, the setting of $\mathcal{R}$ is one of the key factors of the complete data processing pipeline.

In addition, an overly large number of cross-validation trials make the follow-up analysis difficult since a subsequent predictive model training in the second stage filtering can

be more expensive than the computational cost of an association test. The grid search of a number of hyperparameters grows quickly to an unmanageable scale even for a small inclusion of more values into each type of hyperparameters. As a result, the computational cost of each hyperparameter combination should be estimated in line with the available computing resources to decide a proper $\mathcal{R}$.

To balance above factors, it is required to carefully determine a realistic number of cross-validation trials for producing a set of meaningful and stable outputs. In this chapter, we demonstrate that hundreds to thousands of such trials can be handled efficiently with very limited computing resources (e.g. $1\times$ GPU) for the first stage filtering on our breast cancer GWAS study, while the hyperparameters optimisation and discriminative signal extraction in the second stage learning can leverage from a small CPU cluster. Given such results, the scale of the experiment can be generally extended for a larger GWAS dataset with a reasonable amount of computing resources.

5.3.3 Determine the Number of Top SNP-pairs

The number of top $\top$ pairs of list $L^r_{\psi\top}$, generated by a test statistic ψ at a re-sampling trial r, can be as large as one million, shown in Chapter 3. The choice of $\top$ has impact on the resulting robust list $\mathbf{L}_\psi$ in terms of the false positives and cross-validation consistency.

By running re-sampling procedure $\mathcal{R}$ times, we end up with $\mathcal{R}$ lists of SNP-pairs with top $\top$ pairs recorded in every list. One simple way of measuring the robustness is to calculate the frequency of each unique SNP-pair across $\mathcal{R}$ lists and use the consistency as the sorting key. All unique pairs can then be ranked accordingly from the largest to the smallest based on the robustness. After that, a fraction of top pairs can be taken as the robust list $\mathbf{L}_\psi$, whose size of interest specified by the users is denoted as $\intercal$.

As we haven shown in Chapter 3 that such re-sampling consistency of robust SNP-pairs in $\mathbf{L}_\psi$ increases as $\top$ increases in general. By including more pairs with less significance of associations during the calculation of a robust list, the resulting top-ranked robust pairs show better consistency, at the risk of introducing insignificant interactions at each trial. We demonstrate in this chapter that it is also applicable to the breast cancer study here. On the contrary, a small $\top$ in each list $L^r_{\psi\top}$ ensures the selected interactions

with higher degrees of associations with the traits, at the risk of missing less significant interactions but are robust across $\mathcal{R}$ re-sampling trials.

Lastly, the size of $\mathbf{L}_\psi$ influences the follow-up analysis. An excessive long list $\mathbf{L}_\psi$ increases the computational overhead of the follow-up predictive model training and may lead to a degradation of performance for too many features being introduced into the model construction. Also the re-sampling consistency for a large portion of SNP-pairs can be at a low level. With the inclusion of those SNPs, the reliability of a derived predictive model and its prediction performance are questioned. To determine a suitable $\top$, multiple choices can be tested, increasing the complexity of the experiment.

5.3.4 Signal Extraction based on Feature Scores

Once the robust list $\mathbf{L}_\psi$ is determined, the SNPs within the list are used as the input of the next stage filtering for discriminative signal extraction. In this stage, the optimal hyperparameter setting of a statistical model should be selected and used to score features. Given the coefficients, such as the weights, of a derived model, it is expected to have a pathway of constructing a feature score from those coefficients to measure the relative contribution to the classification for each SNP. As we have discussed in Chapter 2 and 4 that multiple metrics are applicable to score a feature's contribution. Some of them are tailored for a network architecture (either shallow or deep in network setting) and there is no gold standard in defining a statistic that is best suitable for all scenarios.

As we already seen in Chapter 4 that, separating the signals and the noise, apart from choosing a suitable feature scoring metric, is not straight forward either. Unlike interpreting a set of scored features for a computer vision application, the choice of a threshold for separating informative SNPs $\mathbf{l}_{\psi\mathbf{P}}$ from the noise set $\mathbf{l}_{\psi\mathbf{N}}$ can not be intuitively judged due to the lack of a ground truth of determining the signal-noise boundary. We have shown that, a simple way could be validating, for each separating threshold of feature scores (all scores are sorted by R_I in the first place and the value at each possible separating position can then be taken as a threshold each time), the prediction performance of the resulting lists $\mathbf{l}_{\psi\mathbf{P}}$ and $\mathbf{l}_{\psi\mathbf{N}}$ respectively and then make a decision based on two validation results. By enumerating all candidates, we expect to find an ideal threshold to separate

the two sets such that the performance of $\mathbf{l}_{\psi\mathbf{P}}$ is close to the one using $\mathbf{L}_{\psi}$ since it contains most informative signals, while the predictive power of $\mathbf{l}_{\psi\mathbf{N}}$ should be quite restricted for a Case-Control binary classification since the noise set contains very little predictors. However, the illustrated validation curves in Chapter 4 are usually not smooth, and it is challenging to propose an ideal procedure to perfectly reduce the false positives in the signal set and the false negatives in the noise set to zero at the same time.

Another consideration is the computational cost of the enumeration process. In each round, a pair of $\mathbf{l}_{\psi\mathbf{P}}$ and $\mathbf{l}_{\psi\mathbf{N}}$ are generated, requiring their respective performance to be evaluated on the validation data. If a training procedure is incurred by a predictive model to evaluate the performance for both sets every time, then the total computational cost is overly expensive for a large number of candidates. Therefore, a light-weight evaluation procedure is needed. Among many methods, if the targeted models are neural nets (such as DCLN), then an efficient procedure can be expected. Given a training solution of a neural net model, which is usually represented by a set of learned weights linking different layers, the features can be scored by interpreting those weights using a specific metric, and then the instances (with the least important features replaced with zeros) in the validation set can be mapped to the decision-making layer to obtain a prediction score for the current round. No expensive re-training occurs for each iteration, and the overhead of traversing through all thresholds is computationally affordable since the cost of such mapping is negligible compared to the heavy model training. As a result, the computational complexity of enumerating all possible thresholds is subject to the input dimension; the sample size of the test set; and the overhead of validation procedure for each round of enumeration.

The previously presented validation procedure essentially relies on the feature scoring metric R_I that provides a lower bound of zero, representing the least important features. If we inspect another one, such as Ω, then there is no such lower bound and the previous threshold enumeration doesn't apply. In this situation, we assume that the scores of noise features fluctuate within a consecutive region of the score spectrum, and the features outside that region are regarded as predictive signals. Therefore, to refine the SNPs allocation to the two sets, it is better to learn the width and location of the noise

region in this situation. Correspondingly, the implementation complexity and computational cost are both increased since a large number of combinations of two factors are to be tested. In conclusion, the way of interpreting the learned weights decided the form of a specific scoring metric, which in turn affects the applicability and efficiency of a potential validation procedure.

In summary, the elementary requirements of a suitable statistical model are to be capable of both properly scoring the features and effectively distinguishing the signals from the noise. In addition, such method is also required to be stable in feature scoring and efficient in performance validation. Having all these characteristics in place, we develop an increased confidence in its signal-noise separation ability.

5.4 Two Stage Filtering Framework

In this section, we introduce a filtering framework to demonstrate that the learning of discriminative SNP-SNP interactions can be achieved with a reasonable amount of computing resources. The entire learning procedure is generally decomposed into two stages of filtering, with their high level work flows depicted in Figure 5.1 and 5.2 respectively.

In stage one, a training set $\mathcal{D}_T$ and a test set $\mathcal{D}_t$ are designated for a given GWAS study. Within the training set, we randomly generate $\mathcal{R}$ model-fitting sets $\mathcal{D}_f^r$ and validation sets $\mathcal{D}_v^r$, for $r = 1, \ldots, \mathcal{R}$. For each $\mathcal{D}_f^r$, an association test ψ is used to perform an exhaustive search via GWISFI and report a list $L_{\psi\top}^r$ of top $\top$ pairs. After running $\mathcal{R}$ trials, each unique SNP-pair is searched across all $\mathcal{R}$ lists to calculate a re-sampling consistency. All SNP-pairs are then sorted from the largest to the smallest based on the consistency, and a fraction of top robust pairs $\mathbf{L}_\psi$ are retained and used in stage two filtering.

In stage two, only the set of SNPs in robust list $\mathbf{L}_\psi$ are kept and the corresponding model-fitting set $\mathcal{D}_f^r(\mathbf{L}_\psi)$, validation set $\mathcal{D}_v^r(\mathbf{L}_\psi)$, training set $\mathcal{D}_T(\mathbf{L}_\psi)$ and test set $\mathcal{D}_t(\mathbf{L}_\psi)$ are created. Next, DCLN is applied with each hyperparameters setting trained on $\mathcal{D}_f^r(\mathbf{L}_\psi)$ and predicted on $\mathcal{D}_v^r(\mathbf{L}_\psi)$. The average performance over all $\mathcal{R}$ cross-validation trials is calculated for each setting and the best one is selected. Once the best hyperparameters setting is decided, we train on the training set and predict on the test set for measuring

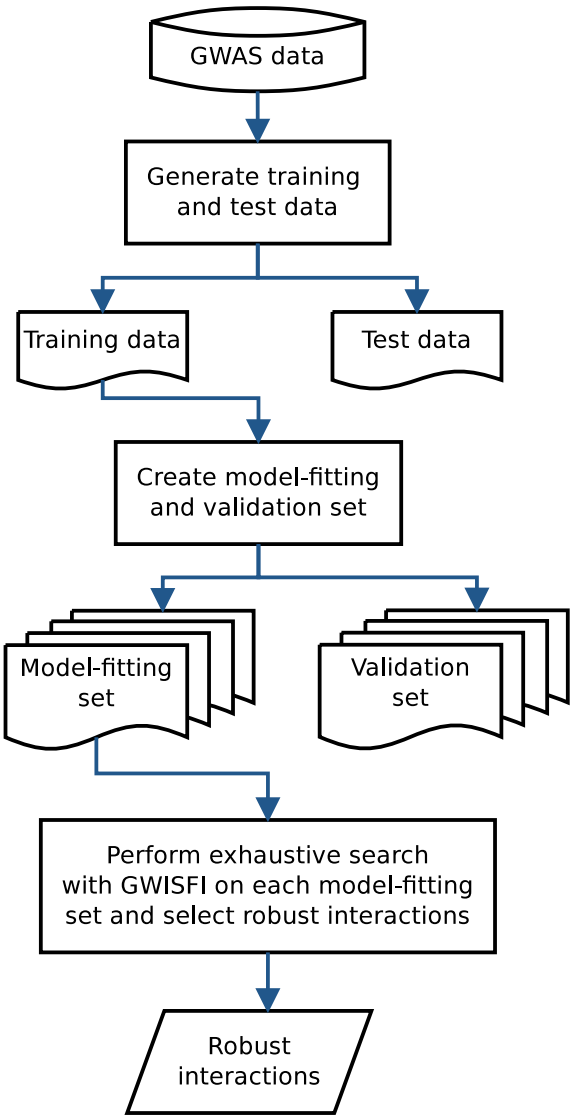

Figure 5.1: High-level overview of stage one filtering.

the prediction performance. Consider the intrinsic randomness in neural-net models, we train DCLN with the best setting on $\mathcal{D}_T(\mathbf{L}_\psi)$ and predict on $\mathcal{D}_t(\mathbf{L}_\psi)$ multiple times (total $\mathcal{R}$ repeated runs) to average the results. The final prediction performance $\overline{Pred(\mathcal{D}_t(\mathbf{L}_\psi))}$ is recorded as a reference. The $\mathcal{R}$ training processes on $\mathcal{D}_T(\mathbf{L}_\psi)$ end up with $\mathcal{R}$ feature scores for each feature. Therefore, we can calculate an average score $\overline{R\_I_i}$ over those repeated trials for each feature x_i. All $\overline{R\_I_i}$ are then sorted from the smallest to the largest. With those sorted scores, we can enumerate all possible thresholds to choose a suitable one, with its basic idea introduced in Chapter 4.

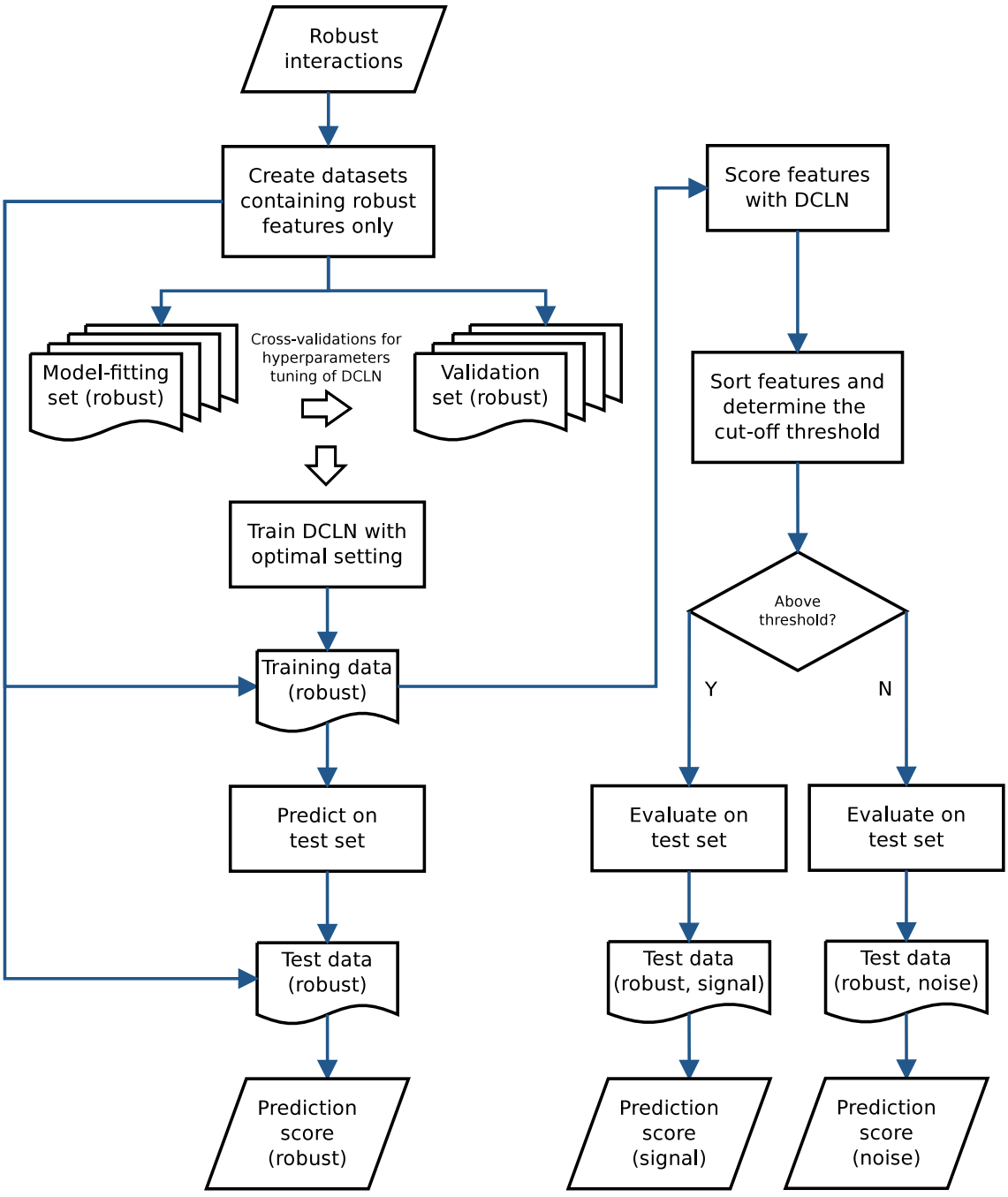

Figure 5.2: High-level overview of stage two filtering.

For each round of enumeration, the lists $\mathbf{l}_{\psi\mathbf{N}}$ and $\mathbf{l}_{\psi\mathbf{P}}$ are initialised. In the first round, set the first smallest score (least important one) as the threshold, allocate SNPs with scores below or equal to the threshold to list $\mathbf{l}_{\psi\mathbf{N}}$ and leave the rest to $\mathbf{l}_{\psi\mathbf{P}}$. After that,

the prediction performance of $\mathbf{l}_{\psi\mathbf{N}}$ and $\mathbf{l}_{\psi\mathbf{P}}$ can be evaluated by mapping the validation data $\mathcal{D}_v^r(\mathbf{l}_{\psi\mathbf{N}})$ and $\mathcal{D}_v^r(\mathbf{l}_{\psi\mathbf{P}})$ to the hidden layer of DCLN to calculate the performance $Pred(\mathcal{D}_v^r(\mathbf{l}_{\psi\mathbf{N}}))$ and $Pred(\mathcal{D}_v^r(\mathbf{l}_{\psi\mathbf{P}}))$. After $\mathcal{R}$ trials of validation, the average performance $\overline{Pred(\mathcal{D}_v(\mathbf{l}_{\psi\mathbf{N}}))}$, $\overline{Pred(\mathcal{D}_v(\mathbf{l}_{\psi\mathbf{P}}))}$ and $\overline{Pred(\mathcal{D}_v(\mathbf{l}_{\psi\mathbf{P}}))} - \overline{Pred(\mathcal{D}_v(\mathbf{l}_{\psi\mathbf{N}}))}$ are calculated for the current threshold. For the next round, the second smallest score is designated as the threshold and the above procedure is repeated. After enumerating all possible thresholds, we select the largest possible one, satisfying $\overline{Pred(\mathcal{D}_v(\mathbf{l}_{\psi\mathbf{P}}))} - \overline{Pred(\mathcal{D}_v(\mathbf{l}_{\psi\mathbf{N}}))} \geq (1-\lambda)\max\left(\overline{Pred(\mathcal{D}_v(\mathbf{l}_{\psi\mathbf{P}}))} - \overline{Pred(\mathcal{D}_v(\mathbf{l}_{\psi\mathbf{N}}))}\right)$, as the solution. This is almost the same as what we have seen in Chapter 4 except for the validation data being used here are a collection of validation sets generated by R trials of re-samplings. Other evaluation criteria are also possible, but we don't wish to complicate the topic here.

Once the best threshold is determined, we allocate SNPs with their average feature scores greater than the threshold to the signal set $\mathbf{l}_{\psi\mathbf{P}}$ and leave the remaining SNPs to the noise set $\mathbf{l}_{\psi\mathbf{N}}$. To validate the predictive power, we evaluate the performance on $\mathcal{D}_t(\mathbf{l}_{\psi\mathbf{P}})$ by prediction mapping, and end up with $\overline{Pred(\mathcal{D}_t(\mathbf{l}_{\psi\mathbf{P}}))}$ (averaged over $\mathcal{R}$ repeated runs). Similarly, the prediction performance $\overline{Pred(\mathcal{D}_t(\mathbf{l}_{\psi\mathbf{N}}))}$ for the noise set can be obtained in such a way. We expect $\overline{Pred(\mathcal{D}_t(\mathbf{l}_{\psi\mathbf{P}}))} \approx \overline{Pred(\mathcal{D}_t(\mathbf{L}_{\psi}))}$, while $\overline{Pred(\mathcal{D}_t(\mathbf{l}_{\psi\mathbf{N}}))}$ should be restricted for binary classification. One thing to note here is that, the performance validation by prediction mapping relies on previously generated training solutions: $\mathbf{w}^*$, $\mathbf{v}_{\mathcal{A}}^*$ and $\mathbf{v}_{\mathcal{B}}^*$. Thus, all such solutions are kept as intermediate results before the end of the experiment. The detailed protocol of the two-stage filtering is given in 4.

5.5 Experiment for Breast Cancer GWAS

5.5.1 Breast Cancer GWAS Data

In this section, we perform the analysis for CIDR Genome-Wide Association Study of Breast Cancer in High Risk Women[1]. The obtained GWAS dataset contains a total number of 4763767 SNPs and 7295 samples (of which 37 subjects are originated from HapMap

[1]This study was funded by a grant CA165038 to Christopher Haiman (University of Southern California) and John Hopper (University of Melbourne) from the National Cancer Institute, National Institute of Health.

Algorithm 4 Interaction features learning by two-stage filtering.

Require: Training set $\mathcal{D}_T$ and test set $\mathcal{D}_t$, rows for samples and columns for SNPs.

Set $\mathcal{R}$, $\top$ and the top number of robust pairs $\intercal$. Choose a test ψ for GWISFI. Set $\mathcal{P}_f$ for the model-fitting set. Set DCLN hyperparameters search space.
for $r \in \{1, \cdots, \mathcal{R}\}$ **do**
  Shuffle the rows of $\mathcal{D}_T$. Take $\mathcal{P}_f$ of $\mathcal{D}_T$ as $\mathcal{D}_f^r$, leaving $\mathcal{P}_v = 1 - \mathcal{P}_f$ of $\mathcal{D}_T$ as $\mathcal{D}_v^r$.
  Run test ψ via GWISFI on $\mathcal{D}_f^r$ and generate a list $L_{\psi\top}^r$ of top $\top$ pairs of SNPs.
end for
Concatenate $\mathcal{R}$ lists $L_{\psi\top}^r$ into a single one; sort all unique SNP-pairs from the largest to the smallest by the frequency (or consistency); and take top $\intercal$ robust pairs as $\mathbf{L}_\psi$.
for $r \in \{1, \cdots, \mathcal{R}\}$ **do**
  Use the set of SNPs in $\mathbf{L}_\psi$ to create $\mathcal{D}_f^r(\mathbf{L}_\psi)$ and $\mathcal{D}_v^r(\mathbf{L}_\psi)$ respectively.
  Grid search for all hyperparameter combinations with each one trained on $\mathcal{D}_f^r(\mathbf{L}_\psi)$ and tested on $\mathcal{D}_v^r(\mathbf{L}_\psi)$. Record the solution $\{\mathbf{w}_f^r, \mathbf{v}_{\mathcal{A}f}^r, \mathbf{v}_{\mathcal{B}f}^r\}$ for each combination.
end for
Select the best hyperparameters setting with max $\overline{Pred(\mathcal{D}_v(\mathbf{L}_\psi))}$.
for $r \in \{1, \cdots, \mathcal{R}\}$ **do**
  Train DCLN with the best setting on $\mathcal{D}_T(\mathbf{L}_\psi)$, and record the training solution $\{\mathbf{w}_T^r, \mathbf{v}_{\mathcal{A}T}^r, \mathbf{v}_{\mathcal{B}T}^r\}$ and feature score $R\_I_i^r$ for each feature x_i.
end for
Calculate $\overline{R\_I_i}$ for each x_i and then sort scores from the smallest to the largest.
Set $\Delta^* = -\infty$; $\mathbf{l}_{\psi\mathbf{P}}^* \leftarrow \varnothing$; and $\lambda = 0.01$ for instance.
for each sorted score in ascending order except for the largest one **do**
  Set $\mathbf{l}_{\psi\mathbf{P}}, \mathbf{l}_{\psi\mathbf{N}} \leftarrow \varnothing$; threshold $\theta =$ current score.
  for $i \in \{1, \cdots, d\}$ **do**
    if $\overline{R\_I_i} \leq \theta$ **then**
      $\mathbf{l}_{\psi\mathbf{N}} \leftarrow \mathbf{l}_{\psi\mathbf{N}} \cup x_i$.
    else
      $\mathbf{l}_{\psi\mathbf{P}} \leftarrow \mathbf{l}_{\psi\mathbf{P}} \cup x_i$.
    end if
  end for
  Set complementary features $\mathbf{l}_{\psi\mathbf{P}}^{\complement}, \mathbf{l}_{\psi\mathbf{N}}^{\complement} \leftarrow 0$. Set $\mathbf{l}_{\psi\mathbf{P}} \leftarrow \mathbf{l}_{\psi\mathbf{P}} \cup \mathbf{l}_{\psi\mathbf{P}}^{\complement}$ and $\mathbf{l}_{\psi\mathbf{N}} \leftarrow \mathbf{l}_{\psi\mathbf{N}} \cup \mathbf{l}_{\psi\mathbf{N}}^{\complement}$, keeping features in their original orders in $\mathbf{L}_\psi$.
  for $r \in \{1, \cdots, \mathcal{R}\}$ **do**
    Map $\mathcal{D}_v^r(\mathbf{l}_{\psi\mathbf{P}})$ and $\mathcal{D}_v^r(\mathbf{l}_{\psi\mathbf{N}})$ to the hidden layer by $\mathbf{w}_f^r$; compare to $\mathbf{v}_{\mathcal{A}f}^r$ and $\mathbf{v}_{\mathcal{B}f}^r$ for classification; and record $Pred(\mathcal{D}_v^r(\mathbf{l}_{\psi\mathbf{P}}))$ and $Pred(\mathcal{D}_v^r(\mathbf{l}_{\psi\mathbf{N}}))$ respectively.
  end for
  Calculate $\Delta_\theta = \overline{Pred(\mathcal{D}_v(\mathbf{l}_{\psi\mathbf{P}}))} - \overline{Pred(\mathcal{D}_v(\mathbf{l}_{\psi\mathbf{N}}))}$ using the averages over $\mathcal{R}$ trials.
  if $\Delta_\theta > \Delta^*$ **then**
    $\Delta^* = \Delta_\theta$.
  end if
end for
Select the max θ satisfying $\Delta_\theta \geq (1 - \lambda)\Delta^*$, and set $\mathbf{l}_{\psi\mathbf{P}}^* \leftarrow \mathbf{l}_{\psi\mathbf{P}}^\theta$.
for $r \in \{1, \cdots, \mathcal{R}\}$ **do**
  Map $\mathcal{D}_t(\mathbf{l}_{\psi\mathbf{P}}^*)$ to the hidden layer by $\mathbf{w}_T^r$; compare to $\mathbf{v}_{\mathcal{A}T}^r$ and $\mathbf{v}_{\mathcal{B}T}^r$ for classification; and record $Pred(\mathcal{D}_t(\mathbf{l}_{\psi\mathbf{P}}^*))$ for trial r.
end for
Report $\overline{Pred(\mathcal{D}_t(\mathbf{l}_{\psi\mathbf{P}}^*))}$ as the validated performance for informative SNPs.

consortium). After quality control using PLINK software (v1.07) with setting "–geno 0.1 –mind 0.1 –maf 0.05 –hwe 0.001", the remaining SNPs and samples are 1774174 and 7288 respectively. To maintain the experiment at a manageable scale, several key population cohorts are selected and split into Germany, Spain and Utah for analysis. The whole-genome of each cohort is then divided into independent chromosomes, and we focus on chromosomes $1 \sim 20$. The objective of the experiment is to find signals for a given cohort, and predict on another cohort to verify the prediction performance. We perform such cross-cohort prediction for each individual chromosome rather than a whole genome-wide scan because we only have a single Nv Tesla K40c GPU for the first stage filtering. Therefore, the quadratic search space of an exhaustive search is contained within each chromosome, which greatly reduces the overall complexity. The genome-wide search, however, can be easily achieved by adding a couple of more GPUs. In this chapter the purpose is to demonstrate the feasibility of proposed method with existing hardware.

5.5.2 Initial Scan for Signal Regions

Before directly applying the computationally intensive protocol 4 to detect interaction features from each chromosome, we first perform a quick scan for every chromosome to locate those containing predictive signals by a simplified procedure since we are restricted by the available computing resources. Given three cohort datasets, we have 6 training-testing combinations: Germany-Utah, Spain-Utah, Germany-Spain, Utah-Spain, Spain-Germany and Utah-Germany. For each combination, we perform an initial screening for every chromosome as follows:

For each training-testing combination in each chromosome, the training cohort is randomly split into two parts multiple times, where 75% samples are allocated for model-fitting, leaving the rest 25% for validation at each trial. There are total five filters being used for an exhaustive search: χ^2, SHEsisEpi, EPIBLASTER, FastEpistasis and Info Gain. Total 500 trials of exhaustive search are performed for the experiment using GWISFI. For each trial, each filter is applied on the model-fitting set and report top 10000 SNP-pairs. After 500 trials of re-samplings, we end up with 500 lists of SNP-pairs and select top 100 robust pairs with highest re-sampling consistency as the inputs to the next stage filtering.

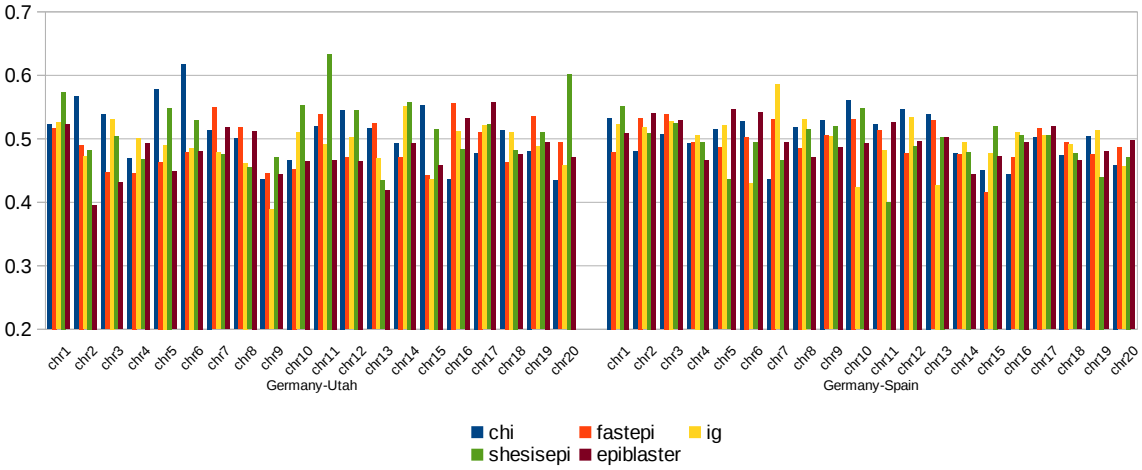

Figure 5.3: Cross-cohorts predictions for each independent chromosome using Germany training set. The balanced accuracy is shown for each case.

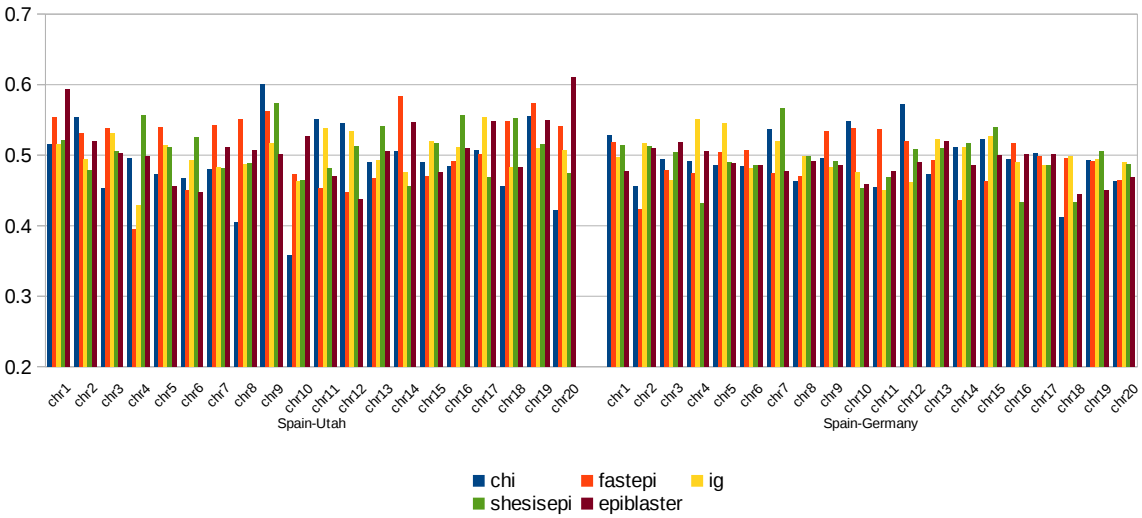

Figure 5.4: Cross-cohorts predictions for each independent chromosome using Spain training set. The balanced accuracy is shown for each case.

For the selected set of robust SNPs, we only keep them and remove others from previously generated 500 model-fitting and validation subsets. For those refined subsets, DCLN is trained on the model-fitting subset for tuning the hyperparameters in a grid-search fashion, with the prediction performance on the validation subset being recorded for each hyperparameter setting. The search space of each hyperparameter is given as follows: $\beta = 100$, $\eta \in \{0.1, 10\}$, $C \in \{0.01, 0.1, 1, 10, 100\}$ and epoch $= 200$, leaving all other settings by default. An average performance over 500 trials is calculated for each

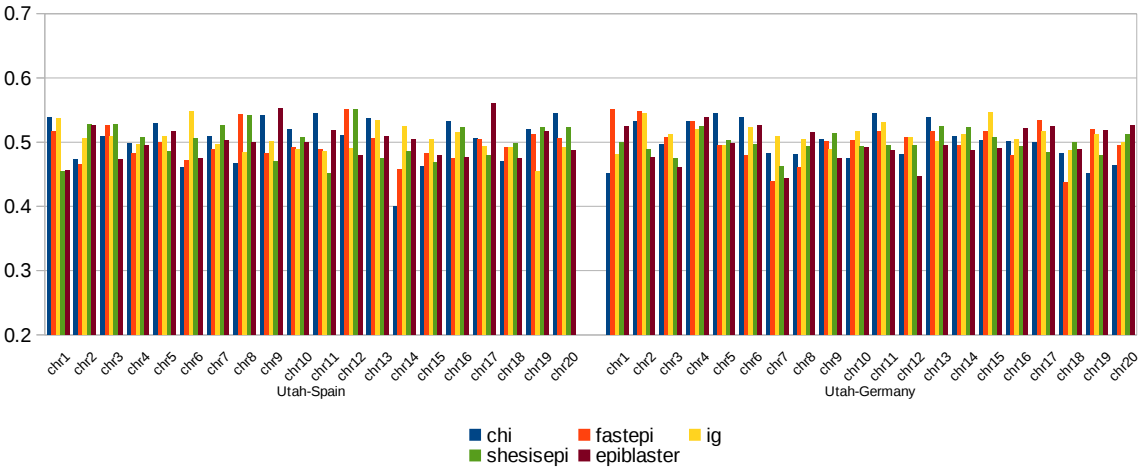

Figure 5.5: Cross-cohorts predictions each independent chromosome using Utah training set. The balanced accuracy is shown for each case.

setting and the best one is selected. Once the best hyperparameters setting is determined, the model is trained on the training cohort and predicted on the testing cohort 500 times to record the average prediction performance. In short, the signal scan at this level uses the robust features for preliminary test of prediction. The outcomes of six cross-cohort predictions across different chromosomes are illustrated in Figure 5.3,5.4 and 5.5 respectively, where the horizontal axis shows the first stage filtering methods used to select robust features for each chromosome and the vertical axis indicates the generated balanced accuracy using robust features.

5.5.3 More Intensive Runs for Key Regions

As can be seen from Figure 5.3 that, predictive signals (we use balanced accuracy > 0.6 as a criterion) appeared in the following regions: 1) chromosome 6 using χ^2 test for first-stage filtering, trained on Germany and predicted on Utah; 2) chromosome 11 and 20 using shesisepi for pre-filtering, trained on Germany and predicted on Utah. From Figure 5.4, the noticeable signals come from chromosome 20 using epiblaster, and chromosome 9 using χ^2 for pre-filtering, trained on Spain and predicted on Utah. We focus on these regions and proceed to more intensive experiments, for which more trials are performed using the complete two stage filtering for prediction evaluation and signal extraction.

The reason of adding more trials is due to several variation factors in the experiment:

- Random sampling of data: The final prediction results are greatly affected by this factor. The exhaustive search by a specific filter selects only a tiny fraction of top features out of a huge volume of candidates. Therefore, it is highly sensitive to the perturbation of data sampling. Although 500× re-samplings are performed for selecting robut SNP-pairs with highest re-sampling consistency, the derived robust list may still be sensitive to the hundred-level re-sampling trials. The random sampling of training data and the nature of an association test jointly decide the robust candidates to be used by the follow-up analysis. Obviously, more repeated sampling trials ease the problem, but the experiment can only operate at a manageable scale, given limited computational resources. This is also the case for DCLN training, for which more trials allow more stable results for hyperparameters tuning during the cross-validation. For the current experiment, there are several training-prediction combinations remained, and we increase the hundred-level re-sampling trials to thousand-level trials for more reliable estimation.
- Randomness of neural-net: The neural-net models contain intrinsic randomness such as the randomly initialised weights, randomly presented instances at a training epoch, etc., and so does the DCLN model. Thus, the result may vary from one trial to the other. More repeated runs for a given pair of training and test sets certainly narrow the confidence interval of estimated prediction performance. It is up to the nature of prediction task and the available computing resources to decide a practical number of repeated runs for a given dataset.

In a word, the false positives and false negatives are generally inevitable. To reduce them and give more reliable estimation, we focus on those key training-testing combinations as described above, and perform the experiment by increasing the total number of re-sampling trials from 500 to 2000, generating a relatively more stable set of robust interactions for predictions.

For the set of SNPs in the robust list, derived from 2000× re-sampling trials, we perform 2000× cross-validations to fine-tune the hyperparameters for DCLN. This time, $\beta = 200$ and epoch $= 400$ are used instead, allowing a more in-depth training. Once

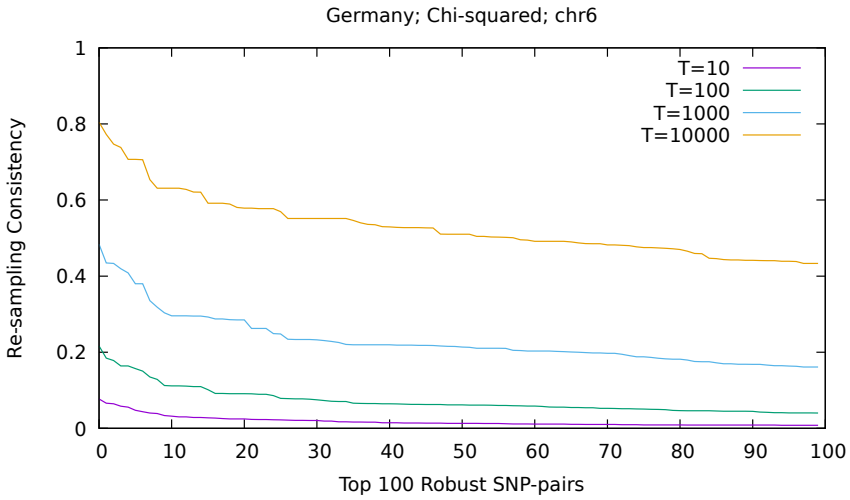

Figure 5.6: Re-sampling consistency for top 100 robust SNP-pairs. Training set: Germany; Filter: χ^2; Chromosome: 6.

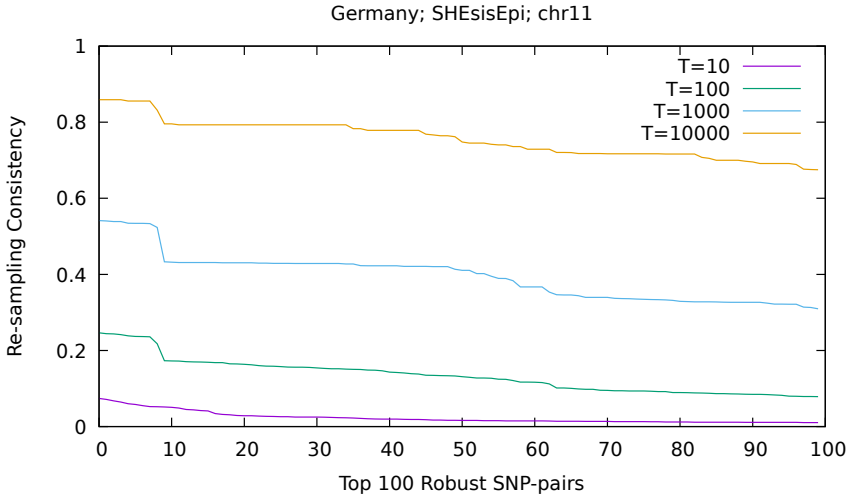

Figure 5.7: Re-sampling consistency for top 100 robust SNP-pairs. Training set: Germany; Filter: *SHEsisEpi*; Chromosome: 11.

the best setting is determined, it is predicted on the test set for performance assessment (train on the training cohort with the best setting and predict on the test cohort for total 2000 times to contain the randomness of neural-net). After this more rigorous experiment, the prediction performance for the combination of Spain-Utah (using χ^2 test for first stage filtering on chromosome 9) falls below 0.6 and is excluded from the next stage analysis. As a result, only four training-testing combinations survived by using the set of SNPs in the robust list. In the next stage, the predictive signal extraction follows the

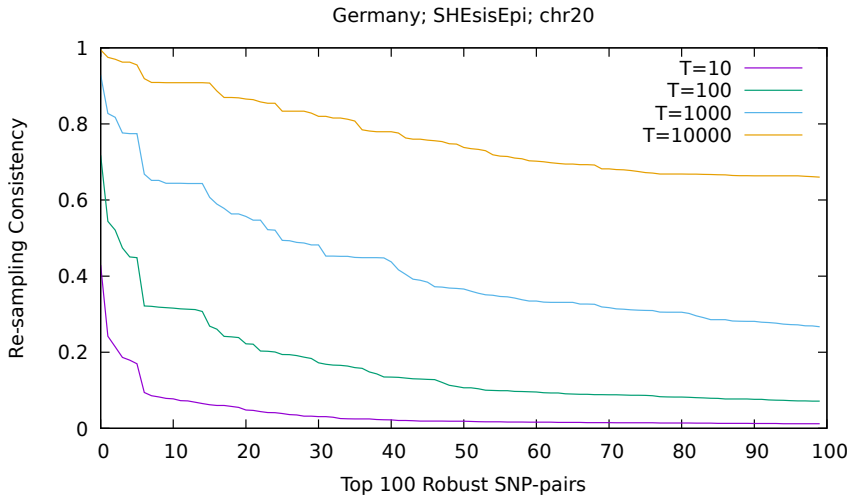

Figure 5.8: Re-sampling consistency for top 100 robust SNP-pairs. Training set: Germany; Filter: *SHEsisEpi*; Chromosome: 20.

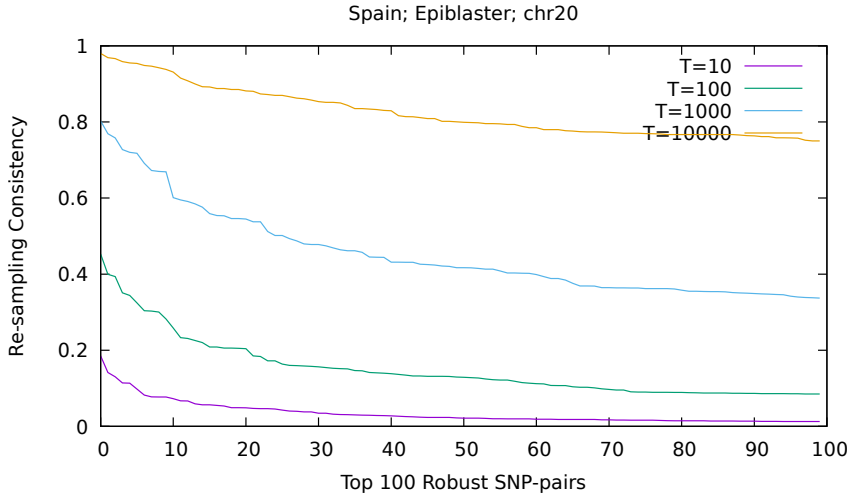

Figure 5.9: Re-sampling consistency for top 100 robust SNP-pairs. Training set: Spain; Filter: *EPIBLASTER*; Chromosome: 20.

steps of the second stage filtering procedure to report results in the end.

One thing to emphasise here again is that the second stage filtering relies on the stable inputs from the first stage filtering. Although an increase to the number of re-sampling trials improves the stability of generated set of robust interactions, the specified top number of pairs $\top$ for each trial also heavily impacted on the derivation of the robust list since it directly affects the re-sampling consistency. Without a sufficient level of robustness, it is hard to believe that the generated SNPs are not false positives, hence we are unable

to expect a reliable prediction on an independent cohort dataset using those SNPs. For different levels of $\top$ used in the experiment, the identified SNP-pairs in the resulting robust list $\mathbf{L}_\psi$ together with their re-sampling consistency may differ greatly for the current study. We only allow the robust SNPs with considerable level of re-sampling consistency to be fed into the second stage filtering.

To give a more clear picture, we plot the re-sampling consistency of top 100 robust SNP-pairs at different levels of $\top \in \{10, 100, 1000, 10000\}$ for each of the remaining four studies in Figure 5.6, 5.7, 5.8 and 5.9 respectively. Obviously, if we only take a very small number of top pairs, say $\top = 10$ for each trial, then the re-sampling consistency of derived SNP-pairs in $\mathbf{L}_\psi$ fall below a low level, and those SNPs are unable to be included into the follow-up analysis. In contrary, the high level consistency can be generally observed for $\top = 10000$, which is adopted by the experiment.

In addition to the DCLN method, we also perform the prediction experiment for the SVM model using robust SNPs as the input features, and the hyperparameters search space for the three SVM kernels are listed as follows:

- SVM Linear: $C \in \{2^{-5}, 2^{-3}, 2^{+3}, 2^{+5}\}$.
- SVM RBF: $C \in \{2^{-5}, 2^{-3}, 2^{+3}, 2^{+5}\}$, $\gamma \in \{2^{-5}, 2^{-3}, 2^{+3}, 2^{+5}\}$.
- SVM Polynomial: $\{2, 3, 4, 5, 6\}$ degrees of polynomials.

Finally, the classification results of two comparing methods are summarised in Table 5.1. From the reported results, DCLN and SVM performed equally well for the first study. In addition, DCLN outperformed SVM for the third study, while SVM reported the best results for the remaining two. Again, DCLN is shown to be highly competitive to the widely-used SVM method.

5.5.4 Discriminative Interactions Extraction

Next, we proceed to the predictive signal extraction for these studies. As it has been shown in the experiment protocol 4 that we are aiming at selecting an optimal threshold θ for the sorted feature scores to separate the predictive SNPs from the noise. Given the determined θ by the protocol, we obtained the signal set $\mathbf{l}_{\psi\mathbf{P}}$ and the noise set $\mathbf{l}_{\psi\mathbf{N}}$ respectively and then validate their respective prediction performance on the test cohort

Table 5.1: Prediction performance comparison between two methods for four breast cancer studies using the set of SNPs in $\mathbf{L}_\psi$ as input features.

Study	**SVM**			**DCLN**	
Germany-Utah; χ^2; chromosome 6	Linear	RBF	Poly	CE	SE
Balanced Accuracy	0.597	0.597	**0.605**	**0.608**	**0.609**
Accuracy	0.598	0.577	0.577	0.600	0.600
AUC	0.672	0.666	0.653	0.662	0.673
Germany-Utah; shesisepi; chromosome 11	Linear	RBF	Poly	CE	SE
Balanced Accuracy	0.605	0.589	**0.620**	0.601	0.585
Accuracy	0.608	0.588	0.608	0.601	0.582
AUC	0.627	0.621	0.631	0.627	0.607
Germany-Utah; shesisepi; chromosome 20	Linear	RBF	Poly	CE	SE
Balanced Accuracy	0.473	0.512	0.523	**0.620**	0.545
Accuracy	0.485	0.505	0.381	0.593	0.545
AUC	0.418	0.523	0.580	0.622	0.541
Spain-Utah; epiblaster; chromosome 20	Linear	RBF	Poly	CE	SE
Balanced Accuracy	**0.629**	0.598	0.567	0.610	0.610
Accuracy	0.567	0.546	0.515	0.568	0.569
AUC	0.687	0.606	0.614	0.619	0.621

by prediction mapping (via the intermediate training solutions stored previously for 2000 trials), and the validation results of the two sets are reported in the second column of Table 5.2. Comparing the balanced accuracy results of the signal sets to the corresponding ones shown in Table 5.1, we can see that the predictive power of the signal sets are well preserved. On the other hand, the validated performance of the noise sets are quite limited for Case-Control classification. From the illustrated validation results, the proposed framework generally meets our expectation.

Additionally, we are also interested in seeing a complete view of every threshold θ (sorted in ascending order) in conjunction with the performance of the corresponding two sets, validated on the test cohort (Utah) for each study. We enumerate all possible thresholds and validate the performance of the two sets by prediction mapping (also via the same training solutions for 2000 trials). The complete enumeration results are shown in Figure 5.10, 5.12, 5.14 and 5.16 respectively for the four studies. We have already seen

Table 5.2: Performance validation results for the signal and noise sets are shown separately for the four studies. Here, Balanced Accuracy* indicates the threshold θ is determined based on the validation sets (using protocol 4), while Balanced Accuracy† is obtained by enumerating all possible thresholds on the test cohort to select the one with largest discrepancy between the signal set and the noise set.

Germany-Utah; χ^2; chr6	Balanced Accuracy*	Balanced Accuracy†	SNPs
Signal	0.607	0.682	36
Noise	0.499	0.499	84
Germany-Utah; shesisepi; chr11	Balanced Accuracy*	Balanced Accuracy†	SNPs
Signal	0.607	0.615	52
Noise	0.351	0.345	5
Germany-Utah; shesisepi; chr20	Balanced Accuracy*	Balanced Accuracy†	SNPs
Signal	0.599	0.662	74
Noise	0.560	0.474	54
Spain-Utah; epiblaster; chr20	Balanced Accuracy*	Balanced Accuracy†	SNPs
Signal	0.628	0.629	59
Noise	0.491	0.452	48

the same type of plots in Chapter 4, with the horizontal axis indicating the threshold θ (sorted from the smallest to the largest) and the vertical axis showing the validation result in balanced accuracy. The purple line represents the results using SNPs in $\mathbf{l}_{\psi\mathbf{P}}$ with feature scores above θ, while the green line represents the performance using SNPs in $\mathbf{l}_{\psi\mathbf{N}}$ with scores below or equal to θ.

The general trends of two curves are similar to what we observed from the computer vision exercises in Chapter 4. The signal set $\mathbf{l}_{\psi\mathbf{P}}$ starts with a high score of performance, and as the threshold increases, less and less SNPs are retained (in other words, more and more least important features in $\mathbf{L}_\psi$ are set to zero, while only above-threshold features are allowed to keep their original values during the prediction mapping) and it reduces to around 0.5 in the end. In contrast, the noise set $\mathbf{l}_{\psi\mathbf{N}}$ starts with a very limited performance number since it only contains the least important feature in the first round of prediction mapping (all other features in $\mathbf{L}_\psi$ are set to zero in that case), and as the threshold increases, more and more features are activated until it reaches to a quality result in

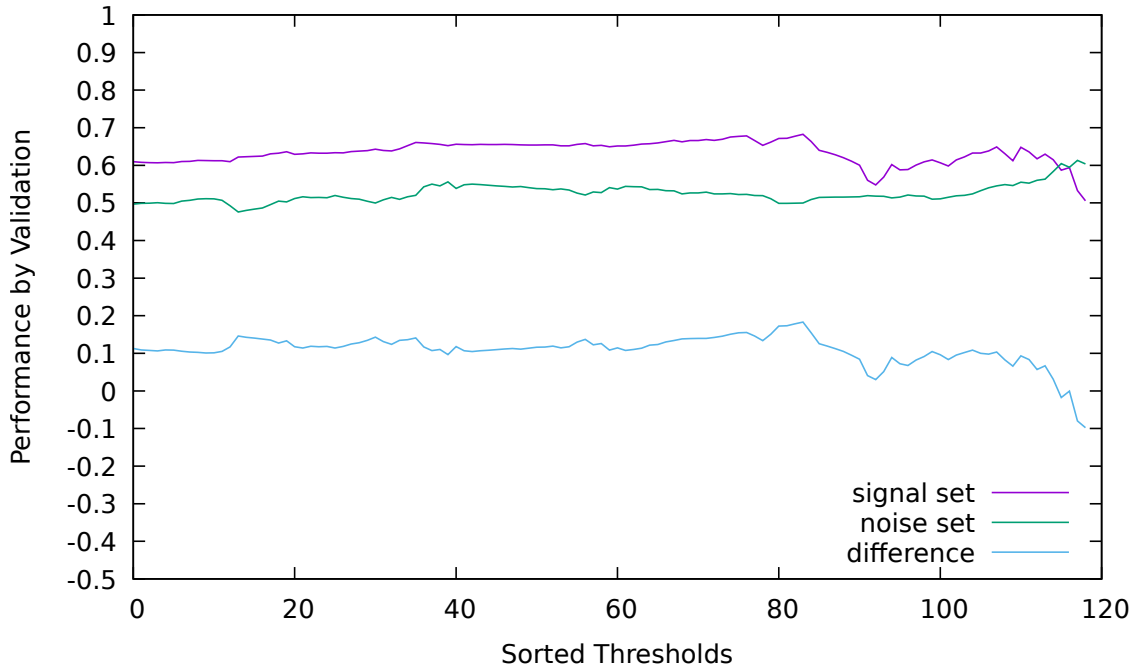

Figure 5.10: Enumeration of θ on the test cohort. Training set: Germany; Test set: Utah; Chromosome: 6; Filter: χ^2 test for robust feature selection.

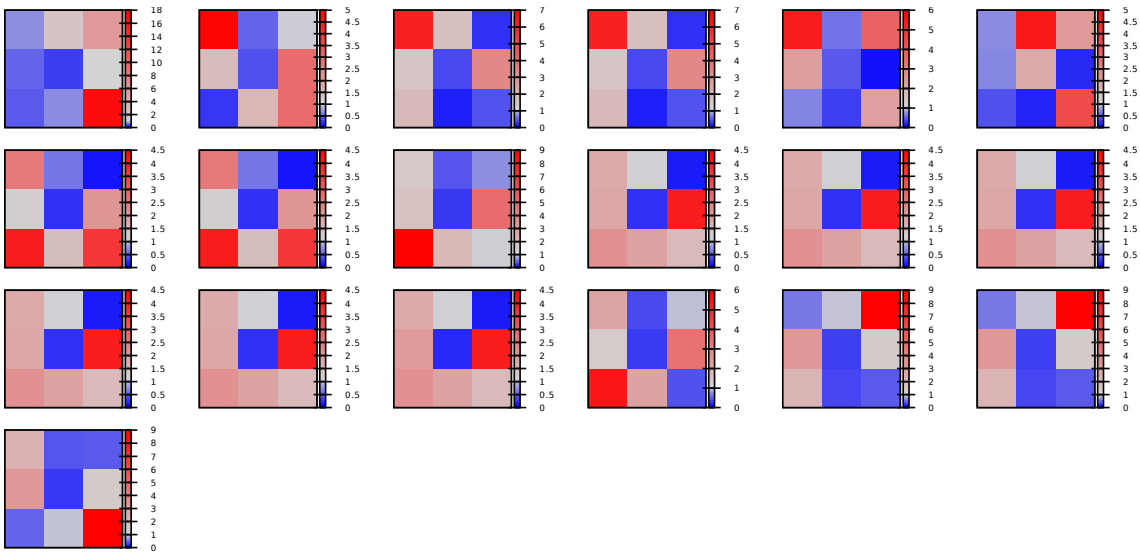

Figure 5.11: Interaction patterns for German Chr6. $\chi^2 + DCLN$ for filtering.

the end. Due to the general curiosity and simplicity, we report, for the threshold θ^{max} that leads to the maximum performance discrepancy between the signals and the noise in prediction mapping, the validation results (in Balanced Accuracy) of $\mathbf{l}_{\psi\mathbf{P}}$ and $\mathbf{l}_{\psi\mathbf{N}}$ in the

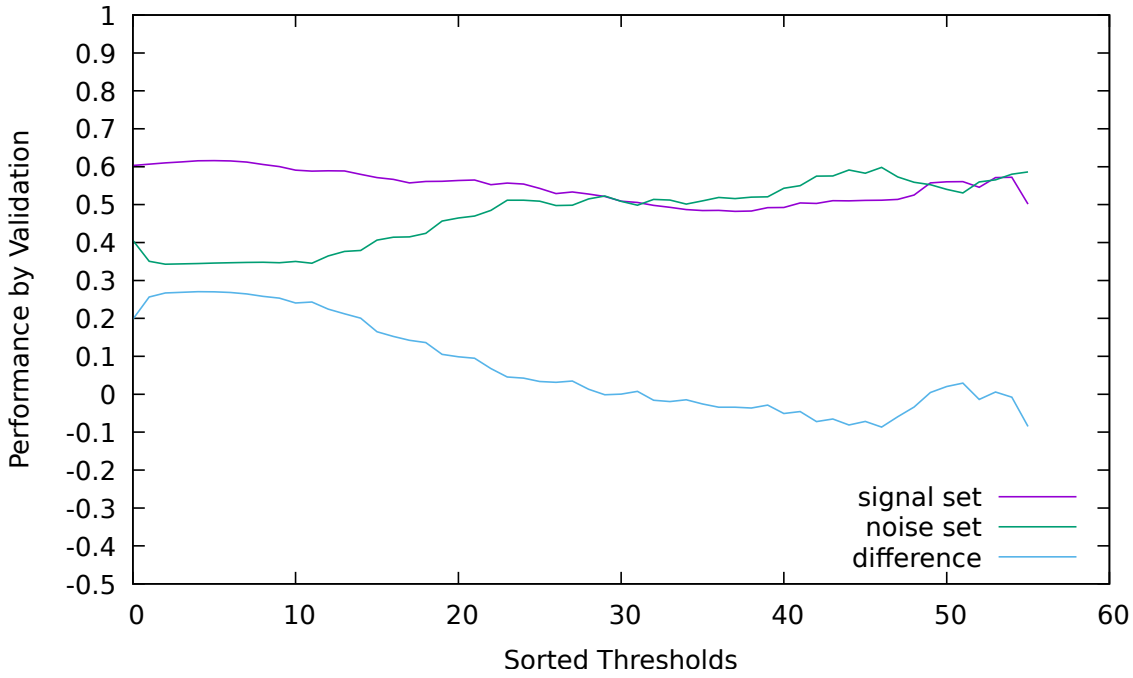

Figure 5.12: Enumeration of θ on the test cohort. Training set: Germany; Test set: Utah; Chromosome: 11; Filter: *SHEsisEpi* test for robust feature selection.

third column of Table 5.2.

Lastly, for the robust interactions in $\mathbf{L}_{\psi}$, only those appearing in $\mathbf{l}_{\psi\mathbf{P}}$ (determined by θ^{max}) are retained for the time being, and we investigate the predictive power externally using SVM. Firstly, for each interaction, we pull out the two SNPs from the same data as the two features for the classification, and perform the training and prediction using SVM to test the performance. Three SVM kernels, namely Linear, RBF and Polynomial, along with their hyperparameters search domain remain the same as before. Each hyperparameter setting of each kernel is used to train on the training set (e.g. German) and predict on the test set (e.g. Utah), with the best result selected for that particular interaction. After the best performance is obtained for each interaction, we focus on those with results ≥ 0.6 and investigate their patterns.

For each generated interaction, we focus on the 3×3 genotype combinations. Firstly a scaling factor $\rho = \#\text{total\_cases}/\#\text{total\_controls}$ is calculated to address the imbalanced sample size for the two classes, and then the counts are collected for each genotype combination n_{ij}^{c} for $i, j \in \{0,1,2\}$, and $c \in \{1,0\}$ representing either Case or Control. After

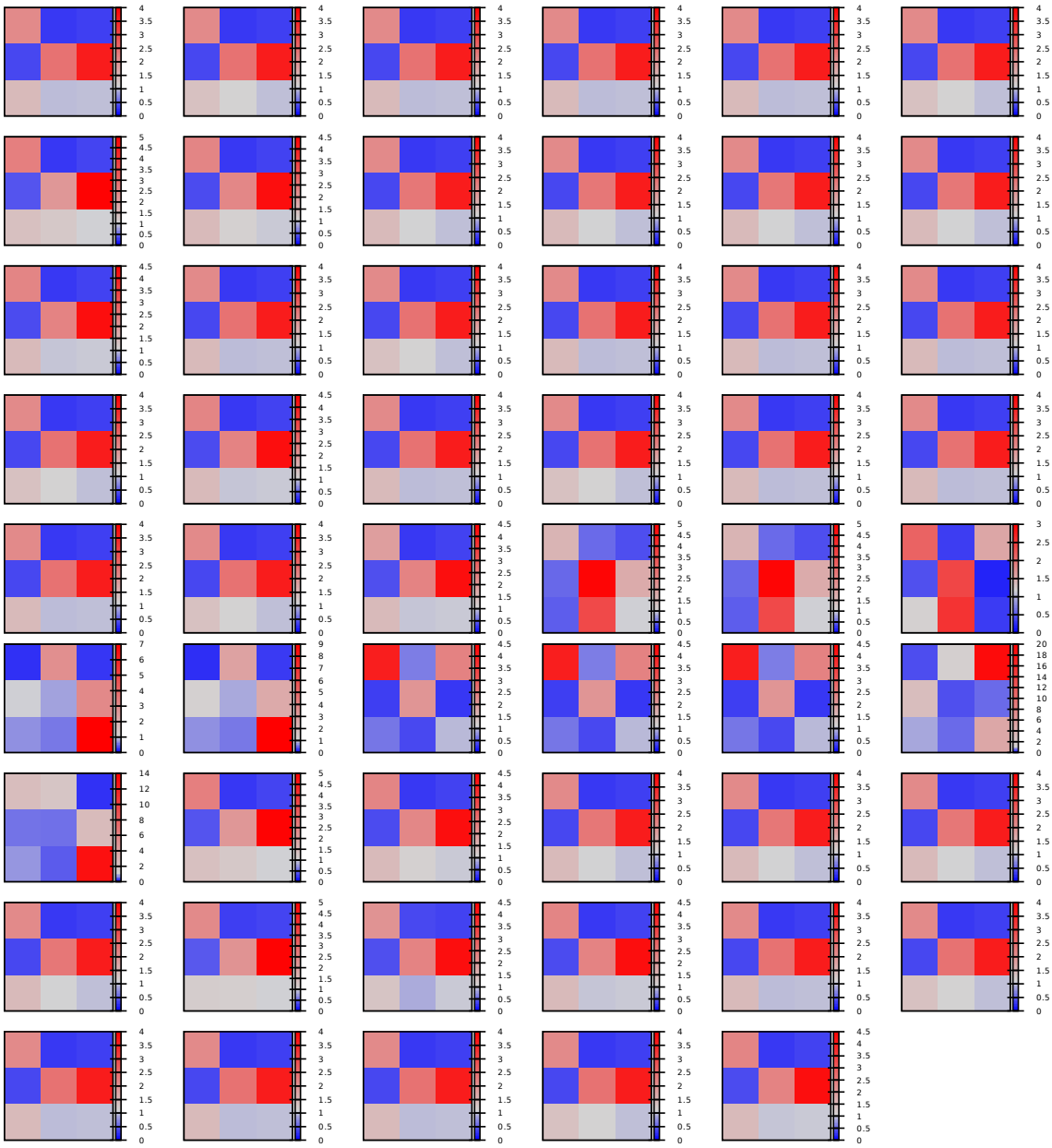

Figure 5.13: Interaction patterns for German Chr11; $SHEsisEpi + DCLN$ for filtering.

that, we calculate the ratio between Cases and Controls for each combination as follows:

$$ratio_{ij} = \frac{n_{ij}^{1} + 0.5}{n_{ij}^{0} \times \rho + 0.5}, \tag{5.1}$$

where a correction 0.5 is appended to each genotype combination for the purpose of

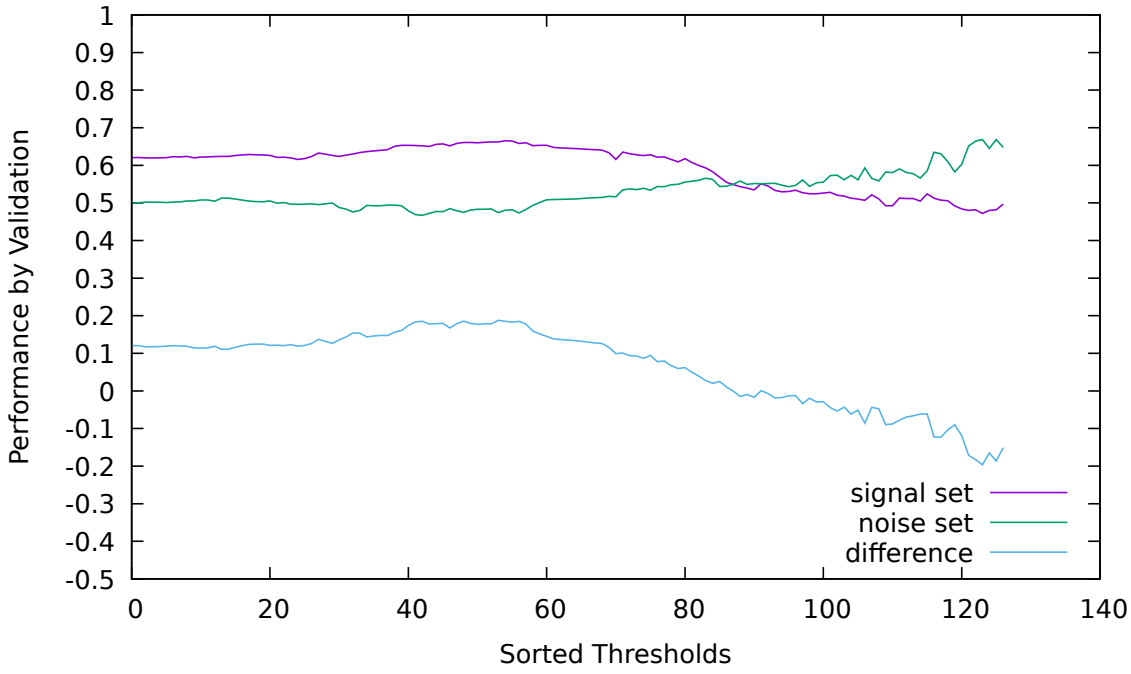

Figure 5.14: Enumeration of θ on the test cohort. Training set: Germany; Test set: Utah; Chromosome: 20; Filter: $SHEsisEpi$ test for robust feature selection.

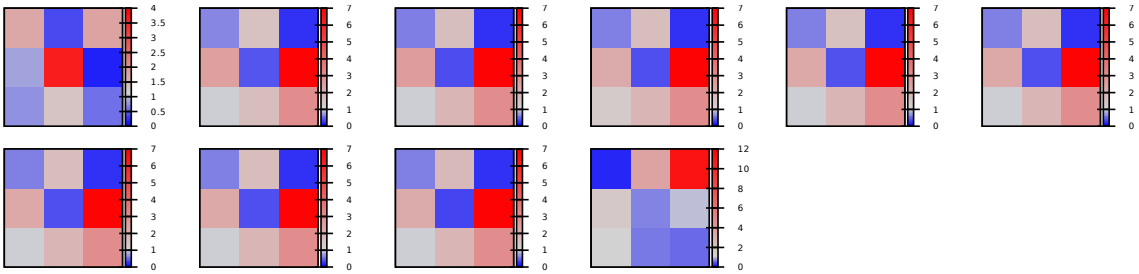

Figure 5.15: Interaction patterns for German Chr20; $SHEsisEpi + DCLN$ for filtering.

dealing with possible empty cells.

Once the ratio is calculated for every combination, a 3×3 ratio table is derived to characterise each interaction pattern. We plot the ratio tables of all discriminative interactions in Figure 5.11, 5.13, 5.15 and 5.17 respectively for the four studies. As can be seen from the plots, the high risk genotype combinations are coloured in red, while the low risk ones are coloured in blue. Different studies varied in interaction patterns and some of them are visually close to nonlinear shapes.

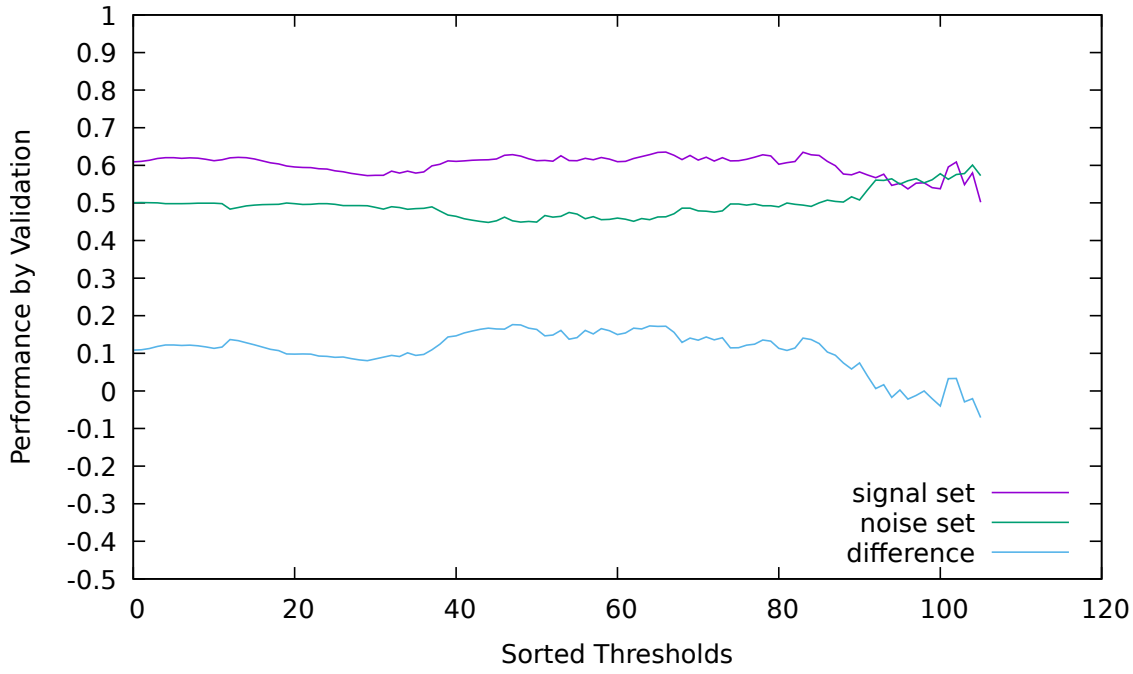

Figure 5.16: Enumeration of θ on the test cohort. Training set: Spain; Test set: Utah; Chromosome: 20; Filter: $EPIBLASTER$ test for robust feature selection.

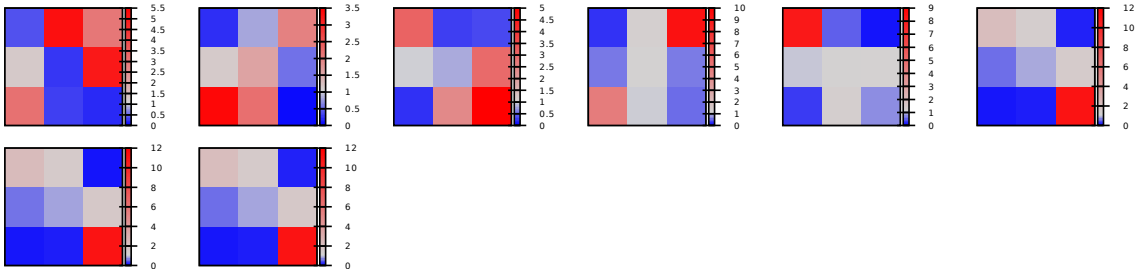

Figure 5.17: Interaction patterns for Spainish Chr20; $EPIBLASTER + DCLN$ for filtering.

5.5.5 Compare to Polygenic Risk Scores

Given a set of SNPs, their joint effects can be summarised as the Polygenic Risk Scores (PRS) for risk prediction of disease [135] on an independent validation dataset. PRS can be calculated as the sum of genotypes, weighted by the effect sizes of the designated SNPs [20, 98], with its simplest form known as [66]:

$$PRS(i) = \sum_{j=1}^{N} \beta_j g_{ij},$$

where g_{ij} is the genotype for SNP j at individual i, and β_j represents the effect size of the SNP, usually given by the GWAS summary statistics [98,135]. In Case-Control binary trait, β_j often takes the form of odds ratio in logarithmic scale [20,89,142]. Once the effect size is obtained, one can quantify the risk score for each individual in the test set, with the prediction accuracy of PRS measured by AUC [66,135,151].

One challenging part of deriving PRS score is to determine SNPs involved in the score calculation in the first place [151]. The traditional way is to first perform GWAS association analysis and then select SNPs with P-values passing a given threshold [66,89,98]. However, this strategy assumes independent SNPs with strong marginal effects, while the desired genuine interactions with little main effects may be missed in such a case, as we have discussed earlier in Chapter 2. Thus, it may generate a search space of SNPs that is undesirable for a further detection of epistatic interactions. For instance, in another breast cancer study [89], SNPs are only tested individually for associations to derive the corresponding PRS, limiting the potential contributions from nonlinear interactions.

Given the PRS, the prediction performance for a number of breast cancer risk studies have been summarised in [151], with limited AUC shown in general. However, it was considered at least 0.7 or a higher AUC is required to reach the minimum level of expected prediction power for practical use [28,122]. Therefore, it is worthwhile checking whether epistatic interactions can supplement to the prediction to some extent. On the other hand, the interaction term can be modeled in multiple ways such as [68,123,142]. By leveraging from bivariate interaction analyses, one such risk score is concluded as a promising complement to the classical PRS [142], while another scoring option outperformed PRS in simulation studies [68].

In contrast, the developed framework adopted bivariate exhaustive search to preselect a number of robust SNPs. In such a case, genuine interactions will be discovered and retained in the first-stage screening if an appropriate statistical test is adopted. The neural-net model subsequently selects predictive signals. In the previous chapter, the model is shown to be flexible to learn various types of interactions with little main effects. For breast cancer studies demonstrated in this chapter, different interaction patterns can be further checked using PRS with embedded interaction terms in future work.

5.6 Summary

In this chapter, the two-stage filtering framework is introduced and applied to breast cancer studies to discover the interaction features for Case-Control classification. The framework combines two proposed techniques to perform feature learning. In the first stage filtering, the re-sampling procedure is adopted to select robust interactions, while the signal selection, hyperparameters tuning and model training are performed at the second stage processing. Cross-cohorts prediction results are reported to demonstrate the performance of the system. If a severe overfitting occurs within those steps, then a quality prediction performance is unrealistic.

To properly analyse the data, different aspects of the experiment setting need to be carefully determined and balanced to enable an effective and practical discovery of discriminative interaction features. From the presented results, it is clear that the robust signals coming out of different chromosomes are given by different test methods in the first stage filtering. These robust interactions become the key to the success of the second stage filtering. The developed tools addressed two main bottlenecks of this type of research, and can be encapsulated into a processing pipeline with other pre-processing, post-processing or intermediate steps to meet the user's research objectives. Overall, the framework identified the desired signal and noise sets, and generally meets our expectation of its practicability for processing real GWAS data.

Chapter 6
Conclusion and Future Work

IN this chapter, we give concluding comments on the proposed research topic and highlight potential future directions. We first give an overview of the proposed research topic, review its scope and importance, and summarise how our contributions address the major technical challenges that are associated with the topic. The limitations and future works are discussed at the end of this chapter.

6.1 Summary of Proposed Research Topic

In this section, we review the research topic in terms of its importance and intrinsically associated challenges.

6.1.1 Importance of Proposed Research Topic

Genome-wide association studies have been drawing a lot of attention in research community. As a novel profiling tool, GWAS is designed to measure single nucleotide polymorphisms (SNPs) in population subsets with complex diseases, such as cancer, in an attempt to explain underlying mechanisms. As the stably inherited genetic variants from generations to generations, SNPs play a central role in revealing the genetic architecture of complex traits and association of genetic variants with disease susceptibility.

Some single-gene diseases such as the Huntington's disease are highly inheritable and subject to the mutations in key independent genes. For those diseases, the single-gene search strategy can identify useful genetic factors, with reported biomarkers used in diagnostic applications. However, for more pervasive complex diseases such as cancers, multiple levels of molecular interactions are non-negligible factors. Most complex diseases follow a non-Mendalian inheritance pattern, where the single-gene strategy doesn't

apply to the disease model. In such a case, it is necessary to consult gene-gene interactions, rather than independent genetic elements with strong marginal effects, to explain possible mechanisms of such diseases.

From the genetic epidemiology point of view, only a small proportion of phenotypic variance can be explained by independent known factors, and gene-gene and gene-environment interactions are hypothesised to significantly contribute to the missing heritability. From the machine learning point of view, on the other hand, the univariate-based feature selection strategy shows little power in detecting epistatic interactions with nonlinear interaction effects. Thus detection of gene-gene interactions becomes an all important goal in our quest to understand genetic architecture of the investigated disease and holds a promise of improved risk assessment and disease prevention.

To address the problem of epistatic interactions detection from a computer science point of view, a desired data processing tool possesses the following characteristics:

- Flexibly approximate various nonlinear patterns at different orders;
- Quickly scan highly associated interactions over a huge search space;
- Detect robust interactions under sampling variations, ensuring the reproducibility;
- Identify epistatic interaction features for effective Case-Control discrimination;
- Perform data processing within a reasonable amount of waiting time.

Due to the technical limitations, the detection of robust and discriminative epistatic interactions faces significant barriers, requiring novel computational tools to satisfy above requirements. The research topic of this thesis is a response to such challenges.

6.1.2 Review of Challenges

The complexity of nonlinear interaction patterns and the high dimensionality of GWAS data are main difficulties faced when attempting to detect and evaluate epistatic interactions. In the ideal case, a highly flexible predictive model can be directly applied to the original data, analyse all possible SNP interactions at different orders, and reliably identify predictive signals for effective Case-Control classification with small computational costs. However this is not realistic as we have discussed in previous chapters.

If the focus is to cover a wide range of search space for association analysis (e.g.

performing an exhaustive search of SNP-interactions at a given order across the entire genome), then a well-defined light-weight statistical test may suffice, at the cost of losing flexibility of learning various nonlinear pattern. This is because a fixed test statistic may be biased to certain types of interaction patterns. Additionally, the predictive power of detected interactions using this strategy is largely unknown.

If the focus is to learn effective predictors for classification, then a nonlinear predictive model is required to be flexible and stable enough to reliably realise different interaction patterns, and be able to effectively distinguish them from the noise. Due to the expensive model-training, the dimensionality of input features has to be reduced. However, the complete search space is invisible to the model.

The overall challenge is how to enable both the search space reduction by fast exhaustive search and effective learning of various interaction patterns by a flexible model. Once the techniques and methodologies are in place, the key to data analysis is to balance their use in the experiment.

Computational Barrier for Association Tests

As we have seen from previous chapters, it is reasonable to achieve this balance by firstly scanning the GWAS data exhaustively for each SNP-pair using tailored statistical tests and then apply the predictive model training on top of the remaining ones. As we have demonstrated previously, even a single run through a medium-sized data incurs a significant computation barrier. Trillion-level tests are usually required for a modern GWAS data. Without dedicated hardware and highly optimised software, weeks and months may be consumed by data processing, which is impractical for research purposes and not suitable for streamlined epidemiological applications.

Various association tests are proposed with each one implemented and optimised in a specific way by a separate research group. Consequently, as we have shown in Chapter 3, the resulting implementations claim various hardware requirements and the reported runtime dramatically differed across the methods. To save precious research and engineering resources, a unified high-performance screening platform is required for easy prototyping different association tests while providing fast exhaustive search capacity.

Via such platform, the exhaustive search is expected to be fast enough that hundreds even thousands of exhaustive runs become feasible for more comprehensive data analyses (e.g. stability assessment) using limited computing resources. An additional benefit of having such platform is to avoid different hardware specifications required by different test methods in literature. Also, methods can work on the same input format and generate outputs that are directly comparable.

Stability in Feature Selection

In bioinformatics research, there are plenty of reported results that are later found to be not replicable on another independent validation dataset. Overfitting is a major cause of this problem, which occurs at all stages of a machine learning procedure. In the feature selection stage, for example, a single exhaustive search of SNP interactions may generate a list of top SNP-pairs with extremely small P-values. Although they are highly significant in association studies, these interactions may be proven to be false positives when variations are introduced to the training data. Thus, the top-ranked interactions by a given trial may vanish in another trial with slightly different training samples.

Obviously, such artefacts should be removed from any follow-up analysis. To overcome this problem, one popular strategy is to use re-sampling procedure to select robust features that consistently appear across multiple sampling trials. We have demonstrated that some methods, such as MDR, have already adopted this strategy. For most statistical tests, however, such re-sampling procedure is impractical since a single exhaustive search may require a long processing time. Repeated re-sampling runs demand an unreasonable amount of compute time. Therefore, there is a need to improve the computational efficiency for providing a pathway to practical re-sampling trials.

Overfitting can also be driven by predictive models that capture the noise in the training data. For example, a common practice for neural networks is to manually stop the training procedure before overfitting to the details of samples, and the stopping criterion can be determined by cross-validation. In the worst case scenario, where the ranks of features for each trial may differ dramatically, the cross-validation may not be able to pick stable features. Thus, a predictive model with intrinsic de-noising capability is required

to avoid this issue and provide relatively stable results over multiple runs.

Model for Learning Predictive Signals

From previous chapters, it is clear that many existing predictive models can be applied to learn epistatic interactions. In theory any nonlinear models are capable of detecting at least some types of nonlinear interactions. But the main issue is that they are mostly black boxes. For example, different kernels give SVM different versions of nonlinear approximation abilities. It is unknown *a priori* which kernels or kernel combinations are suitable for which types of nonlinear patterns at various orders. As we have seen in Chapter 4 that the RBF kernel works well for synthetic data, while in Chapter 5 the polynomial kernel performed the best for most breast cancer studies.

In addition to the universal approximation ability required for flexible learning of different nonlinear patterns, a desired model should also be at least exhibiting some characteristics that are in line with the human's choice, behaviour and cognition to be trustworthy for its discovered features. At the same time it should allow simple procedures to recognise the discriminative signals for classification. This provides some interpretations for what has been learned by the model and whether the realised knowledge makes sense to us. Although it is not possible to pursue a perfect solution for all scenarios, the confidence in the detected features can be increased by approximating such characteristic. Importantly, the method should be stable in feature ranking, otherwise no conclusion can be made from a set of results generated by multiple runs of the model training. Lastly, the method is required to be simple and efficient such that it can be used by an expensive experiment protocol like cross-validation to evaluate the performance.

6.1.3 Summary of Contributions

In response to the challenges, our proposed techniques partially achieved the overall objective as follows.

- To address the stable feature selection using re-sampling trials, we proposed a GPU-based screening platform called GWISFI, enabling the prototyping of cus-

tomised filtering methods and the fast exhaustive search for pairwise interactions at the same time. The platform provides a functional interface for users to implement their own statistical tests, while all exhaustive search-related computations are handled by the system automatically. Significant runtime improvements are observed after re-implementing the original methods via GWISFI, and consequently the re-sampling-based stable interactions selection becomes feasible thanks to the fast processing speed. In addition, the efficiency of the system can be further improved by proposed two extension algorithms for robust interactions selection by re-sampling trials. The main advantages of this piece of work are briefly highlighted as follows:

1. Ultra-fast processing speed for exhaustive search of bivariate interactions using limited GPU resources.
2. Functional interface allowing users to define their own statistical tests using a number of building elements.
3. Practical detection for robust interactions via re-sampling trials. Extension algorithms are provided to further improve the efficiency.

- Neural networks are popular predictive models and are universal approximators. We devised a novel neural network model called DCLN, enabling the binary classification and feature selection at the same time. The model is shallow in nature, thus it can be simply embedded into an expensive experiment protocol for evaluation. The classification ability of this method is shown to be highly competitive to the state-of-the-art method, while its scored features triggered the concept-level understanding and are compatible with the human's cognition, indicating its way of characterising the feature contribution is close to a human's choice. It also allows a light-weight validation procedure to effectively and efficiently distinguish the predictive signals from the noise, making it a practical tool for selecting discriminative features. Its main advantages are highlighted as follows:

  1. Shallow in network architecture. The simplicity of the model eases the understanding of information flow from the input to the hidden layer, compared to more complex architectures such as deep learning models. The reduced

complexity enables relatively more efficient data processing.

2. Better interpretability. The generated feature scores are reasonable and agree with the human's understanding. The feature scores revealed what were learned by DCLN, making it more interpretable and consequently more trustworthy than pure black-box methods.
3. Signal-noise differentiation. Instead of just scoring each feature, DCLN allows a simple validation procedure to actually separate the signals from the noise with minimum computational overhead.
4. No extra engineering effort or computational overhead required for optimising additional factors such as kernel selection (e.g. multiple kernel learning) performed for optimising a kernel machine.
5. Stable in feature ranking. In Chapter 4, the DCLN model is shown to stably identify various nonlinear interactions (and their combinations) under different noise levels for synthetic datasets.
6. Flexible and model-free. Unlike filter methods (e.g. association tests), DCLN is classification-oriented and there is no need to specify the order of interactions or assume certain types of interactions *a priori*.

- By combining two proposed techniques into a two-stage filtering framework, we systematically analyse the breast cancer GWAS data. Because a series of experiment settings are to be determined for each technique, the proposed framework takes different factors into account and balances the use of two methods. The framework is shown to be feasible for data analysis and generally satisfies our expectation of learning epistatic interaction features that are robust and discriminative for effective Case-Control classification using limited computing resources.

6.2 Overall Conclusion

The proposed high-performance detection methods are technical contributions to the GWAS research endevour. These methods are simple by themselves and do not require an access to very large-scale computing infrastructure for the presented studies. From

the experimental results, the learning of robust and discriminative interaction features for effective Case-Control classification is proven to be practical using modern hardware, making it a useful tool for analysing a broad range of Case-Control GWAS datasets.

6.3 Future Work

In this section, we discuss possible future research directions. Although not all of them may have a possible solution, it is worth discussing at least some interesting aspects.

6.3.1 Contingency Table Calculations

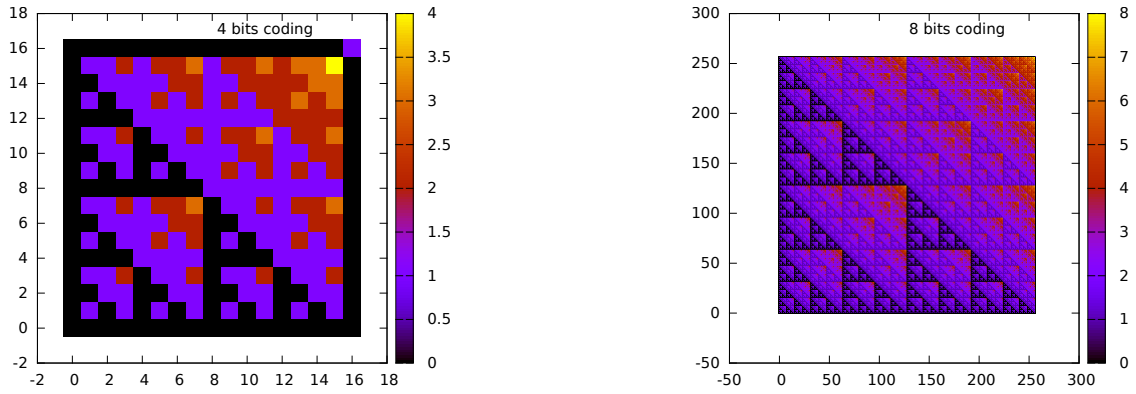

Figure 6.1: Bit-counts for all possible combinations.

As introduced in Chapter 3, the contingency table calculations are accountable for more than $\sim$ 90% of computing time in GPU test. As we know, the major bottlenecks are bitwise AND operation and hardware population count for counting genotype frequency in GPU. If such heavy operations can be substituted by relatively light-weight operations, then a significant saving of processing time would be achieved. To investigate such possibilities, we review the bit-coding scheme of genotype.

Recall each integer word is encoded from 64 bits binary numbers for 64 consecutive instances at a time. Therefore, it represents total 2^{64} possible values. Consequently for each genotype combination in the 3×3 table, there are total $2^{64} \times 2^{64}$ possible frequencies using a pair of integers. One curiosity is to visualise all possible frequencies generated by

all possible pairs of 64-bit words in a 2D plot, using each axis representing all possible 2^{64} values. However, the huge number of possible combinations exceed the limit of plotting. To simplify this operation, we reduce to 8-bit data representation and plot all possible $2^8 \times 2^8$ frequencies in Figure 6.1.

As can be seen, the plot shows fractal regularities. Because of repeated patterns appearing in the plot, one hypothesis is that the frequency of a genotype combination can be expressed directly by a function (the overhead is expected to be much smaller than a bitwise operation) of 2 coordinates from 2 axes. If a closed-form solution exists without incurring a recursive call, then the expensive bitwise operation and population count can be bypassed for each pair of integers. Without sufficient knowledge in fractal research area, we are unable to determine its possible values in solving this problem at this time. We leave it to future work to explore such possibilities.

6.3.2 Feature Scoring in Neural Network

We have shown in Chapter 4 that the scoring metric of informative features is constructed from the learned weights and being a proxy of measuring the feature's relative contribution/importance to the binary classification. In future work, we would like to systematically investigate such metrics together with other possible measurements to provide a solid theoretical foundation for revealing the underlying mechanisms. Due to the simplicity of DCLN method, the possible understanding of such mechanisms should be relatively easier than a model with a complex architecture. In short, the scored features reveal what knowledge is learned by the model and whether such knowledge is reasonable to a binary classification problem. Although the interpretability is improved, we still wish to know why and how the learning procedure generates such information.

On the other hand, we use cross-validations for fine-tuning the hyperparameters during the experiment. In future work, we would like to investigate the relation between the hyperparameter settings and the resulting feature scores in the fine-tuning stage. With some levels of such understanding, each hyperparameter search space may be better determined in the first place such that the search space for all hyperparameter combinations is very likely to be reduced.

6.3.3 Higher Order Interactions

One of the most challenging goals in statistical genetics is elucidation of higher order interactions that involve more than two interacting loci. The complexity and scale of this problem are overly large such that there are no satisfactory methods to attack this problem currently. In such a case, the exhaustive search inevitably becomes impractical unless significant simplifications can be made to the computation. Theoretically the detection for discriminative high-order interactions shouldn't be a problem for proposed neural network model. For instance, we have demonstrated its prediction capacity using synthetic third-order interaction datasets in Chapter 4, but in future we will test its feature selection ability for such scenarios using various types of synthetic data followed by investigation of real life genotyping data of complex diseases. Lastly, current research topic purely focuses on gene-gene interactions, which are considered important to reveal the disease aetiology. It is also important to mention that gene-environment interaction undoubtedly contribute in a major way to disease risk, and availability of computationally efficient framework, such as the one developed in the course of this work, may prove to be essential for further investigation in this area.

Bibliography

[1] A. Agresti, *An Introduction to Categorical Data Analysis*, 2nd ed. Hoboken, New Jersey: Wiley-Interscience, 2007.

[2] S. Alelyani, H. Liu, and L. Wang, "The effect of the characteristics of the dataset on the selection stability," in *2011 IEEE 23rd International Conference on Tools with Artificial Intelligence*, 2011, pp. 970–977.

[3] D. H. Alexander and K. Lange, "Stability selection for genome-wide association," *Genetic Epidemiology*, vol. 35, no. 7, pp. 722–728, 2011. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/gepi.20623

[4] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan, "Multiple kernel learning, conic duality, and the smo algorithm," in *Proceedings of the Twenty-First International Conference on Machine Learning*, ser. ICML '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 6. [Online]. Available: https://doi.org/10.1145/1015330.1015424

[5] A. J. Banegas-Luna, B. Imbernón, A. L. Castro, A. Pérez-Garrido, J. P. Cerón-Carrasco, S. Gesing, I. Merelli, D. D'Agostino, and H. Pérez-Sánchez, "Advances in distributed computing with modern drug discovery," *Expert Opinion on Drug Discovery*, vol. 14, no. 1, pp. 9–22, 2019, pMID: 30484337. [Online]. Available: https://doi.org/10.1080/17460441.2019.1552936

[6] A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera, "Explainable artificial intelligence (xai): Concepts, taxonomies,

opportunities and challenges toward responsible ai," *Information Fusion*, vol. 58, pp. 82–115, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253519308103

[7] W. Bateson, *Mendel's Principles of Heredity*. Cambridge University Press, 1909.

[8] J. Bedő, D. Rawlinson, B. Goudey, and C. S. Ong, "Stability of bivariate gwas biomarker detection," *PLOS ONE*, vol. 9, no. 4, pp. 1–17, 2014. [Online]. Available: https://doi.org/10.1371/journal.pone.0093319

[9] A. Bhat, P. R. Lucek, and J. Ott, "Analysis of complex traits using neural networks," *Genetic Epidemiology*, vol. 17, no. S1, pp. S503–S507, 1999. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/gepi.1370170781

[10] C. M. Bishop, *Pattern recognition and machine learning*. Singapore: Springer Science+Business Media, LLC, 2006.

[11] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, ser. COLT '92. New York, NY, USA: Association for Computing Machinery, 1992, p. 144–152. [Online]. Available: https://doi.org/10.1145/130385.130401

[12] D. Brinza, M. Schultz, G. Tesler, and V. Bafna, "RAPID detection of gene–gene interactions in genome-wide association studies," *Bioinformatics*, vol. 26, no. 22, pp. 2856–2862, 09 2010. [Online]. Available: https://doi.org/10.1093/bioinformatics/btq529

[13] L. Briollais *et al.*, "Methodological issues in detecting gene-gene interactions in breast cancer susceptibility: a population-based study in ontario," *BMC Medicine*, vol. 5, no. 1, p. 22, 2007.

[14] K. H. Brodersen, C. S. Ong, K. E. Stephan, and J. M. Buhmann, "The balanced accuracy and its posterior distribution," in *2010 20th International Conference on Pattern Recognition*, 2010, pp. 3121–3124.

[15] X. Cao, G. Yu, W. Ren, M. Guo, and J. Wang, "Dualwmdr: Detecting epistatic interaction with dual screening and multifactor dimensionality reduction," *Human Mutation*, vol. 41, no. 3, pp. 719–734, 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/humu.23951

[16] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, May 2011, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm. [Online]. Available: https://doi.org/10.1145/1961189.1961199

[17] C. C. Chang, C. C. Chow, L. C. Tellier, S. Vattikuti, S. M. Purcell, and J. J. Lee, "Second-generation PLINK: rising to the challenge of larger and richer datasets," *GigaScience*, vol. 4, no. 1, 02 2015, s13742-015-0047-8. [Online]. Available: https://doi.org/10.1186/s13742-015-0047-8

[18] C. Chang, L. Rampásek, and A. Goldenberg, "Dropout feature ranking for deep learning models," *CoRR*, vol. abs/1712.08645, 2017. [Online]. Available: http://arxiv.org/abs/1712.08645

[19] L. Chen, G. Yu, D. J. Miller, L. Song, C. Langefeld, D. Herrington, Y. Liu, and Y. Wang, "A ground truth based comparative study on detecting epistatic snps," in *2009 IEEE International Conference on Bioinformatics and Biomedicine Workshop*, 2009, pp. 26–31.

[20] S. W. Choi, T. S.-H. Mak, and P. F. O'Reilly, "Tutorial: a guide to performing polygenic risk score analyses," *Nature Protocols*, vol. 15, no. 9, pp. 2759–2772, 2020.

[21] Y. Chung, S. Y. Lee, R. C. Elston, and T. Park, "Odds ratio based multifactor-dimensionality reduction method for detecting gene–gene interactions," *Bioinformatics*, vol. 23, no. 1, pp. 71–76, 11 2006. [Online]. Available: https://doi.org/10.1093/bioinformatics/btl557

[22] H. J. Cordell, "Detecting gene-gene interactions that underlie human diseases," *Nature Reviews Genetics*, vol. 10, no. 6, pp. 392–404, 2009.

[23] H. J. Cordell, "Epistasis: what it means, what it doesn't mean, and statistical methods to detect it in humans," *Human Molecular Genetics*, vol. 11, no. 20, pp. 2463–2468, 10 2002. [Online]. Available: https://doi.org/10.1093/hmg/11.20.2463

[24] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[25] R. Culverhouse, B. K. Suarez, J. Lin, and T. Reich, "A perspective on epistasis: limits of models displaying no main effect," *Am J Hum Genet*, vol. 70, no. 2, pp. 461–471, 2002.

[26] C. A. Davis, F. Gerick, V. Hintermair, C. C. Friedel, K. Fundel, R. Küffner, and R. Zimmer, "Reliable gene signatures for microarray classification: assessment of stability and performance," *Bioinformatics*, vol. 22, no. 19, pp. 2356–2363, 07 2006. [Online]. Available: https://doi.org/10.1093/bioinformatics/btl400

[27] F. Dorani and T. Hu, "Feature selection for detecting gene-gene interactions in genome-wide association studies," in *Applications of Evolutionary Computation*, K. Sim and P. Kaufmann, Eds. Cham: Springer International Publishing, 2018, pp. 33–46.

[28] F. Dudbridge, "Power and predictive accuracy of polygenic risk scores," *PLOS Genetics*, vol. 9, no. 3, pp. 1–17, 03 2013. [Online]. Available: https://doi.org/10.1371/journal.pgen.1003348

[29] D. M. Evans, J. Marchini, A. P. Morris, and L. R. Cardon, "Two-stage two-locus models in genome-wide association," *PLOS Genetics*, vol. 2, no. 9, pp. 1–9, 09 2006. [Online]. Available: https://doi.org/10.1371/journal.pgen.0020157

[30] P. Fergus, C. C. Montañez, B. Abdulaimma, P. Lisboa, C. Chalmers, and B. Pineles, "Utilizing deep learning and genome wide association studies for epistatic-driven preterm birth classification in african-american women," *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, vol. 17, no. 2, p. 668–678, Mar. 2020. [Online]. Available: https://doi.org/10.1109/TCBB.2018.2868667

[31] R. A. Fisher, "The correlations between relatives on the supposition of mendelian inheritance," *Philosophical Transactions of the Royal Society of Edinburgh*, vol. 52, pp. 399–433, 1918.

[32] S. B. Gabriel *et al.*, "Segregation at three loci explains familial and population risk in hirschsprung disease," *Nature Genetics*, vol. 31, no. 1, pp. 89–93, 2002.

[33] M. H. Gail, "Discriminatory Accuracy From Single-Nucleotide Polymorphisms in Models to Predict Breast Cancer Risk," *JNCI: Journal of the National Cancer Institute*, vol. 100, no. 14, pp. 1037–1041, 07 2008. [Online]. Available: https://doi.org/10.1093/jnci/djn180

[34] Y. Gal, J. Hron, and A. Kendall, "Concrete dropout," in *NIPS*, 2017.

[35] A. Ghorbani, A. Abid, and J. Zou, "Interpretation of neural networks is fragile," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 3681–3688, Jul. 2019. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/4252

[36] J. González-Domínguez and B. Schmidt, "Gpu-accelerated exhaustive search for third-order epistatic interactions in case–control studies," *Journal of Computational Science*, vol. 8, pp. 93–100, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877750315000393

[37] B. Goudey, D. Rawlinson, Q. Wang, F. Shi, H. Ferra, R. M. Campbell, L. Stern, M. T. Inouye, C. S. Ong, and A. Kowalczyk, "Gwis - model-free, fast and exhaustive search for epistatic interactions in case-control gwas," *BMC Genomics*, vol. 14, no. 3, p. S10, 2013.

[38] C. S. Greene, N. A. Sinnott-Armstrong, D. S. Himmelstein, P. J. Park, J. H. Moore, and B. T. Harris, "Multifactor dimensionality reduction for graphics processing units enables genome-wide testing of epistasis in sporadic ALS," *Bioinformatics*, vol. 26, no. 5, pp. 694–695, 01 2010. [Online]. Available: https://doi.org/10.1093/bioinformatics/btq009

[39] J. Gui, A. S. Andrew, P. Andrews, H. M. Nelson, K. T. Kelsey, M. R. Karagas, and J. H. Moore, "A robust multifactor dimensionality reduction method for detecting gene–gene interactions with application to the genetic analysis of bladder cancer susceptibility," *Annals of Human Genetics*, vol. 75, no. 1, pp. 20–28, 2011. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1469-1809.2010.00624.x

[40] J. Gui, J. H. Moore, S. M. Williams, P. Andrews, H. L. Hillege, P. van der Harst, G. Navis, W. H. Van Gilst, F. W. Asselbergs, and D. Gilbert-Diamond, "A simple and computationally efficient approach to multifactor dimensionality reduction analysis of gene-gene interactions for quantitative traits," *PLOS ONE*, vol. 8, no. 6, pp. 1–7, 06 2013. [Online]. Available: https://doi.org/10.1371/journal.pone.0066545

[41] I. Guyon, J. Weston, and S. Barnhill, "Gene selection for cancer classification using support vector machines," *Machine Learning*, vol. 46, no. 1, pp. 389–422, 2002.

[42] A. Gyenesei, J. Moody, A. Laiho, C. A. Semple, C. S. Haley, and W.-H. Wei, "BiForce Toolbox: powerful high-throughput computational analysis of gene–gene interactions in genome-wide association studies," *Nucleic Acids Research*, vol. 40, no. W1, pp. W628–W632, 06 2012. [Online]. Available: https://doi.org/10.1093/nar/gks550

[43] M. Gönen and E. Alpaydın, "Multiple kernel learning algorithms," *Journal of Machine Learning Research*, vol. 12, pp. 2211–2268, 07 2011.

[44] I. B. Hallgrímsdóttir and D. S. Yuster, "A complete classification of epistatic two-locus models," *BMC Genetics*, vol. 9, no. 1, p. 17, 2008.

[45] J. Han and C. Moraga, "The influence of the sigmoid function parameters on the speed of backpropagation learning," in *From Natural to Artificial Neural Computation*, J. Mira and F. Sandoval, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 195–201.

[46] Z. He and W. Yu, "Stable feature selection for biomarker discovery," *Computational Biology and Chemistry*, vol. 34, no. 4, pp. 215–225, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1476927110000502

[47] J. N. Hirschhorn and M. J. Daly, "Genome-wide association studies for common diseases and complex traits," *Nat Rev Genet*, vol. 6, no. 2, pp. 95–108, 2005.

[48] J. N. Hirschhorn, "Genomewide association studies — illuminating biologic pathways," *New England Journal of Medicine*, vol. 360, no. 17, pp. 1699–1701, 2009, pMID: 19369661. [Online]. Available: https://doi.org/10.1056/NEJMp0808934

[49] J. Hoh and J. Ott, "Mathematical multi-locus approaches to localizing complex human trait genes," *Nature Reviews Genetics*, vol. 4, no. 9, pp. 701–709, 2003.

[50] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0893608089900208

[51] D. W. Hosmer and S. Lemeshow, *Applied logistic regression*. New York: John Wiley & Sons, 2000.

[52] T. Hu, Y. Chen, J. W. Kiralis, R. L. Collins, C. Wejse, G. Sirugo, S. M. Williams, and J. H. Moore, "An information-gain approach to detecting three-way epistatic interactions in genetic association studies," *Journal of the American Medical Informatics Association*, vol. 20, no. 4, pp. 630–636, 02 2013. [Online]. Available: https://doi.org/10.1136/amiajnl-2012-001525

[53] T. Hu, Y. Chen, J. W. Kiralis, and J. H. Moore, "Visen: Methodology and software for visualization of statistical epistasis networks," *Genetic Epidemiology*, vol. 37, no. 3, pp. 283–285, 2013. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/gepi.21718

[54] X. Hu *et al.*, "Shesisepi, a gpu-enhanced genome-wide snp-snp interaction scanning algorithm, efficiently reveals the risk genetic epistasis in bipolar disorder," *Cell Research*, vol. 20, no. 7, pp. 854–857, 2010.

[55] Huan Liu and Lei Yu, "Toward integrating feature selection algorithms for classification and clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 4, pp. 491–502, 2005.

[56] J. Jakobsdottir, M. B. Gorin, Y. P. Conley, R. E. Ferrell, and D. E. Weeks, "Interpretation of genetic association studies: Markers with replicated highly significant odds ratios may be poor classifiers," *PLOS Genetics*, vol. 5, no. 2, pp. 1–8, 02 2009. [Online]. Available: https://doi.org/10.1371/journal.pgen.1000337

[57] A. C. J. Janssens and C. M. van Duijn, "Genome-based prediction of common diseases: advances and prospects," *Human Molecular Genetics*, vol. 17, no. R2, pp. R166–R173, 10 2008. [Online]. Available: https://doi.org/10.1093/hmg/ddn250

[58] W. Joubert, J. Nance, S. Climer, D. Weighill, and D. Jacobson, "Parallel accelerated custom correlation coefficient calculations for genomics applications," *Parallel Computing*, vol. 84, pp. 15–23, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167819118301431

[59] W. Joubert, J. Nance, D. Weighill, and D. Jacobson, "Parallel accelerated vector similarity calculations for genomics applications," *Parallel Computing*, vol. 75, pp. 130–145, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016781911830084X

[60] L. Kaderali, "Neural network feature selection in complex trait analysis," *Technical Report, ZAIK, University of Cologne*, 2003.

[61] A. Kalousis, J. Prados, and M. Hilario, "Stability of feature selection algorithms: a study on high-dimensional spaces," *Knowledge and Information Systems*, vol. 12, no. 1, pp. 95–116, 2007.

[62] T. Kam-Thong *et al.*, "Epiblaster-fast exhaustive two-locus epistasis detection strategy using graphical processing units," *European Journal of Human Genetics*, vol. 19, no. 4, pp. 465–471, 2011.

[63] T. Kam-Thong *et al.*, "Glide: Gpu-based linear regression for detection of epistasis," *Human Heredity*, vol. 73, no. 4, pp. 220–236, 2012.

[64] J. R. Koza and J. P. Rice, "Genetic generation of both the weights and architecture for a neural network," in *IJCNN-91-Seattle International Joint Conference on Neural Networks*, vol. ii, 1991, pp. 397–404 vol.2.

[65] P. Kraft *et al.*, "Beyond odds ratios - communicating disease risk based on genetic profiles," *Nature Reviews Genetics*, vol. 10, no. 4, pp. 264–269, 2009.

[66] S. A. Lambert, G. Abraham, and M. Inouye, "Towards clinical utility of polygenic risk scores," *Human Molecular Genetics*, vol. 28, no. R2, pp. R133–R142, 07 2019. [Online]. Available: https://doi.org/10.1093/hmg/ddz187

[67] Q. V. Le, M. Ranzato, R. Monga, M. Devin, K. Chen, G. S. Corrado, J. Dean, and A. Y. Ng, "Building high-level features using large scale unsupervised learning," in *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ser. ICML'12. Madison, WI, USA: Omnipress, 2012, pp. 507–514.

[68] T. Le., H. Gong., P. Orzechowski., E. Manduchi., and J. Moore, "Expanding polygenic risk scores to include automatic genotype encodings and gene-gene interactions," in *Proceedings of the 13th International Joint Conference on Biomedical Engineering Systems and Technologies - BIOINFORMATICS*, INSTICC. SciTePress, 2020, pp. 79–84.

[69] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[70] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[71] H. Lee, A. Battle, R. Raina, and A. Y. Ng, "Efficient sparse coding algorithms," in *NIPS*, 2017.

[72] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: Association for Computing Machinery, 2009, pp. 609–616. [Online]. Available: https://doi.org/10.1145/1553374.1553453

[73] C. M. Lewis and J. Knight, "Introduction to genetic association studies," *Cold Spring Harbor Protocols*, vol. 2012, no. 3, pp. 297–306, 2012. [Online]. Available: http://cshprotocols.cshlp.org/content/2012/3/pdb.top068163.abstract

[74] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," *ACM Comput. Surv.*, vol. 50, no. 6, Dec. 2017. [Online]. Available: https://doi.org/10.1145/3136625

[75] W. Li and J. Reich, "A complete enumeration and classification of two-locus disease models," *Human Heredity*, vol. 50, no. 6, pp. 334–349, 2000.

[76] Y. Li, C.-Y. Chen, and W. W. Wasserman, "Deep feature selection: Theory and application to identify enhancers and promoters," *Journal of Computational Biology*, vol. 23, no. 5, pp. 322–336, 2016, pMID: 26799292. [Online]. Available: https://doi.org/10.1089/cmb.2015.0189

[77] Z. C. Lipton, "The mythos of model interpretability," *Queue.*, vol. 16, no. 3, pp. 31–57, 2018.

[78] Z. Lipton, "The mythos of model interpretability," *Communications of the ACM*, vol. 61, 10 2016.

[79] Y. Liu, W. Duan, J. Paschall, and N. L. Saccone, "Artificial neural networks for linkage analysis of quantitative gene expression phenotypes and evaluation of gene x gene interactions," *BMC Proceedings*, vol. 1, no. 1, p. S47, 2007.

[80] Y. Y. Lu, Y. Fan, J. Lv, and W. S. Noble, "Deeppink: reproducible feature selection in deep neural networks," in *NIPS*, 2018.

[81] P. R. Lucek, J. Hanke, J. Reich, S. A. Solla, and J. Ott, "Multi-locus nonparametric linkage analysis of complex trait loci with neural networks," *Hum Hered*, vol. 48, no. 5, pp. 275–284, 1998.

[82] P. R. Lucek and J. Ott, "Neural network analysis of complex traits," *Genetic Epidemiology*, vol. 14, no. 6, pp. 1101–1106, 1997. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291098-2272%281997%2914%3A6%3C1101%3A%3AAID-GEPI90%3E3.0.CO%3B2-K

[83] N. Luíza da Costa, M. Dias de Lima, and R. Barbosa, "Evaluation of feature selection methods based on artificial neural network weights," *Expert Systems with Applications*, vol. 168, p. 114312, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417420310083

[84] B. Maher, "Personal genomes: The case of the missing heritability," *Nature*, vol. 456, no. 7218, pp. 18–21, November 2008. [Online]. Available: https://doi.org/10.1038/456018a

[85] T. A. Manolio *et al.*, "Finding the missing heritability of complex diseases," *Nature*, vol. 461, no. 7265, pp. 747–753, 2009.

[86] J. Marchini, P. Donnelly, and L. R. Cardon, "Genome-wide strategies for detecting multiple loci that influence complex diseases," *Nature Genetics*, vol. 37, no. 4, pp. 413–417, 2005.

[87] M. Marinov and D. E. Weeks, "The complexity of linkage analysis with neural networks," *Hum Hered*, vol. 51, no. 3, pp. 169–176, 2001.

[88] N. Matchenko-Shimko and M. P. Dube, "Gene-gene interaction tests using svm and neural network modeling," in *2007 IEEE Symposium on Computational Intelligence and Bioinformatics and Computational Biology*, 2007, pp. 90–97.

[89] N. Mavaddat *et al.*, "Polygenic risk scores for prediction of breast cancer and breast cancer subtypes," *The American Journal of Human Genetics*, vol. 104, no. 1, pp. 21–34, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0002929718304051

[90] J. H. McDonald, *Handbook of Biological Statistics*, 3rd ed. Baltimore: Sparky House Publishing, 2014.

[91] M. L. McHugh, "The chi-square test of independence," *Biochemia Medica*, vol. 23, no. 2, pp. 143–149, 2013.

[92] B. A. McKinney, D. M. Reif, M. D. Ritchie, and J. H. Moore, "Machine learning for detecting gene-gene interactions: a review," *Appl Bioinformatics*, vol. 5, no. 2, pp. 77–88, 2006.

[93] R. S. Michalski, "A theory and methodology of inductive learning," *Artificial Intelligence*, vol. 20, no. 2, pp. 111–161, 1983. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0004370283900164

[94] J. Mielniczuk and M. Rdzanowski, "Use of information measures and their approximations to detect predictive gene-gene interaction," *Entropy*, vol. 19, no. 1, pp. 1–23, 2017. [Online]. Available: https://www.mdpi.com/1099-4300/19/1/23

[95] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K. R. Mullers, "Fisher discriminant analysis with kernels," in *Neural Networks for Signal Processing IX: Proceedings of the 1999 IEEE Signal Processing Society Workshop (Cat. No.98TH8468)*, 1999, pp. 41–48.

[96] M. C. Mills and C. Rahal, "A scientometric review of genome-wideassociation studies," *Communications Biology*, vol. 2, no. 1, p. 9, 2019.

[97] J. Minnier, M. Yuan, J. S. Liu, and T. Cai, "Risk classification with an adaptive naive bayes kernel machine model," *Journal of the American Statistical Association*, vol. 110, no. 509, pp. 393–404, 2015. [Online]. Available: https://doi.org/10.1080/01621459.2014.908778

[98] S. Mistry *et al.*, "The use of polygenic risk scores to identify phenotypes associated with genetic risk of bipolar disorder and depression: A systematic review," *Journal of Affective Disorders*, vol. 234, pp. 148–155, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016503271732373X

[99] J. H. Moore, "A global view of epistasis," *Nature Genetics*, vol. 37, no. 1, pp. 13–14, 2005.

[100] J. H. Moore and S. M. Williams, "Epistasis and its implications for personal genetics," *The American Journal of Human Genetics*, vol. 85, no. 3, pp. 309–320, 2009.

[101] J. H. Moore, C. S. Greene, P. C. Andrews, and B. C. White, "Does complexity matter? artificial evolution, computational evolution and the genetic analysis of epistasis in common human diseases," in *Genetic Programming Theory and Practice VI*. Boston, MA: Springer US, 2009, pp. 1–19. [Online]. Available: https://doi.org/10.1007/978-0-387-87623-8_9

[102] J. H. Moore and M. Sipper, "Grammatical evolution strategies for bioinformatics and systems genomics," in *Handbook of Grammatical Evolution*, C. Ryan, M. O'Neill, and J. Collins, Eds. Cham: Springer International Publishing, 2018, pp. 395–405. [Online]. Available: https://doi.org/10.1007/978-3-319-78717-6_16

[103] J. A. Morris and M. J. Gardner, "Statistics in medicine: Calculating confidence intervals for relative risks (odds ratios) and standardised ratios and rates," *BMJ*, vol. 296, no. 6632, pp. 1313–1316, 1988. [Online]. Available: https://www.bmj.com/content/296/6632/1313

[104] A. A. Motsinger, S. M. Dudek, L. W. Hahn, and M. D. Ritchie, "Comparison of neural network optimization approaches for studies of human genetics," in *Applications of Evolutionary Computing*, F. Rothlauf, J. Branke, S. Cagnoni, E. Costa, C. Cotta, R. Drechsler, E. Lutton, P. Machado, J. H. Moore, J. Romero, G. D. Smith, G. Squillero, and H. Takagi, Eds., vol. 3907. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 103–114.

[105] A. A. Motsinger-Reif, S. M. Dudek, L. W. Hahn, and M. D. Ritchie, "Comparison of approaches for machine-learning optimization of neural networks for detecting gene-gene interactions in genetic epidemiology," *Genetic Epidemiology*, vol. 32, no. 4, pp. 325–340, 2008. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/gepi.20307

[106] C. Niel, C. Sinoquet, C. Dina, and G. Rocheleau, "A survey about methods dedicated to epistasis detection," *Frontiers in Genetics*, vol. 6, p. 285, 2015. [Online]. Available: https://www.frontiersin.org/article/10.3389/fgene.2015.00285

[107] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, no. 6583, pp. 607–609, 1996.

[108] H. Pearson, "What is a gene?" *Nature*, vol. 441, no. 7092, pp. 398–401, 2006.

[109] S. Purcell, "Plink 1.07," *http://pngu.mgh.harvard.edu/purcell/plink/*, 2007.

[110] S. Purcell *et al.*, "Plink: a tool set for whole-genome association and population-based linkage analyses," *American journal of human genetics*, vol. 81, no. 3, pp. 559–575, 2007.

[111] M. Ranzato, Y.-L. Boureau, and Y. LeCun, "Sparse feature learning for deep belief networks," in *NIPS*, 2007.

[112] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 1135–1144. [Online]. Available: https://doi.org/10.1145/2939672.2939778

[113] N. Risch and K. Merikangas, "The future of genetic studies of complex human diseases," *Science*, vol. 273, no. 5281, pp. 1516–1517, 1996. [Online]. Available: https://science.sciencemag.org/content/273/5281/1516

[114] M. D. Ritchie, B. C. White, J. S. Parker, L. W. Hahn, and J. H. Moore, "Optimization of neural network architecture using genetic programming improves detection and modeling of gene-gene interactions in studies of human diseases," *BMC Bioinformatics*, vol. 4, p. 28, 2003.

[115] M. Ritchie, L. Hahn, N. Roodi, L. Bailey, W. Dupont, F. Parl, and J. Moore, "Multifactor-dimensionality reduction reveals high-order interactions among estrogen-metabolism genes in sporadic breast cancer," *American Journal of Human Genetics*, vol. 69, no. 1, pp. 138–147, 2001.

[116] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by backpropagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[117] N. L. Saccone, T. J. Downey Jr., D. J. Meyer, R. J. Neuman, and J. P. Rice, "Mapping genotype to phenotype for linkage analysis," *Genetic Epidemiology*, vol. 17, no. S1, pp. S703–S708, 1999. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/gepi.13701707115

[118] Y. Saeys, I. Inza, and P. Larrañaga, "A review of feature selection techniques in bioinformatics," *Bioinformatics*, vol. 23, no. 19, pp. 2507–2517, 08 2007. [Online]. Available: https://doi.org/10.1093/bioinformatics/btm344

[119] H. Sanz, C. Valim, E. Vegas, J. M. Oller, and F. Reverter, "Svm-rfe: selection and visualization of the most relevant features through non-linear kernels," *BMC Bioinformatics*, vol. 19, no. 1, p. 432, 2018.

[120] S. M. Schmutz and T. G. Berryere, "Genes affecting coat colour and pattern in domestic dogs: a review," *Animal Genetics*, vol. 38, no. 6, pp. 539–549, 2007. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1365-2052.2007.01664.x

[121] D. Segrè, A. DeLuna, G. M. Church, and R. Kishony, "Modular epistasis in yeast metabolism," *Nature Genetics*, vol. 37, no. 1, pp. 77–83, 2005.

[122] P. D. Shah, "Polygenic Risk Scores for Breast Cancer—Can They Deliver on the Promise of Precision Medicine?" *JAMA Network Open*, vol. 4, no. 8, pp. e2 119 333–e2 119 333, 08 2021. [Online]. Available: https://doi.org/10.1001/jamanetworkopen.2021.19333

[123] B. Sheppard, N. Rappoport, P.-R. Loh, S. J. Sanders, N. Zaitlen, and A. Dahl, "A model and test for coordinated polygenic epistasis in complex traits," *Proceedings of the National Academy of Sciences*, vol. 118, no. 15, 2021. [Online]. Available: https://www.pnas.org/content/118/15/e1922305118

[124] J. Siemiatycki and D. C. Thomas, "Biological Models and Statistical Interactions: an Example from Multistage Carcinogenesis," *International Journal of Epidemiology*, vol. 10, no. 4, pp. 383–387, 12 1981. [Online]. Available: https://doi.org/10.1093/ije/10.4.383

[125] G. F. Smits and E. M. Jordaan, "Improved svm regression using mixtures of kernels," in *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No.02CH37290)*, vol. 3, 2002, pp. 2785–2790.

[126] S. Sonnenburg, G. Rätsch, and C. Schäfer, "A general and efficient multiple kernel learning algorithm," in *NIPS*, 2005.

[127] L. Squarcina, U. Castellani, and P. Brambilla, "Chapter 8 - multiple kernel learning," in *Machine Learning*, A. Mechelli and S. Vieira, Eds. Academic Press, 2020, pp. 141–156. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128157398000080

[128] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[129] V. Tam *et al.*, "Benefits and limitations of genome-wide association studies," *Nature Reviews Genetics*, vol. 20, no. 8, pp. 467–484, 2019.

[130] J. Tang, S. Alelyani, and H. Liu, "Feature selection for classification: A review," in *Data Classification*. CRC Press, Jan. 2014, pp. 37–64.

[131] J. W. Templeton, A. P. Stewart, and W. S. Fletcher, "Coat color genetics in the Labrador retriever," *Journal of Heredity*, vol. 68, no. 2, pp. 134–136, 03 1977. [Online]. Available: https://doi.org/10.1093/oxfordjournals.jhered.a108792

[132] The Wellcome Trust Case-Control Consortium, "Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls," *Nature*, vol. 447, no. 7145, pp. 661–678, 2007.

[133] S.-F. Tsai, C.-W. Tung, C.-A. Tsai, and C.-T. Liao, "An exhaustive scan method for snp main effects and snp x snp interactions over highly homozygous genomes," *Journal of Computational Biology*, vol. 24, no. 12, pp. 1254–1264, 2017, pMID: 29099245. [Online]. Available: https://doi.org/10.1089/cmb.2017.0140

[134] S. Tuo, H. Liu, and H. Chen, "Multipopulation harmony search algorithm for the detection of high-order SNP interactions," *Bioinformatics*, vol. 36, no. 16, pp. 4389–4398, 03 2020. [Online]. Available: https://doi.org/10.1093/bioinformatics/btaa215

[135] E. Uffelmann *et al.*, "Genome-wide association studies," *Nature Reviews Methods Primers*, vol. 1, no. 1, p. 59, 2021.

[136] R. Upstill-Goddard, D. Eccles, J. Fliege, and A. Collins, "Machine learning approaches for the discovery of gene–gene interactions in disease data," *Briefings in Bioinformatics*, vol. 14, no. 2, pp. 251–260, 05 2012. [Online]. Available: https://doi.org/10.1093/bib/bbs024

[137] R. J. Urbanowicz, J. Kiralis, N. A. Sinnott-Armstrong, T. Heberling, J. M. Fisher, and J. H. Moore, "Gametes: a fast, direct algorithm for generating pure, strict, epistatic models with random architectures," *BioData Mining*, vol. 5, no. 1, p. 16, 2012.

[138] P. M. Visscher, "Sizing up human height variation," *Nature Genetics*, vol. 40, no. 5, pp. 489–490, 2008.

[139] P. M. Visscher *et al.*, "10 years of gwas discovery: Biology, function, and translation," *Am J Hum Genet*, vol. 101, no. 1, pp. 5–22, 2017.

[140] P. M. Visscher, W. G. Hill, and N. R. Wray, "Heritability in the genomics era - concepts and misconceptions," *Nature Reviews Genetics*, vol. 9, no. 4, pp. 255–266, 2008.

[141] X. Wan, C. Yang, Q. Yang, H. Xue, X. Fan, N. L. S. Tang, and W. Yu, "Boost: A fast approach to detecting gene-gene interactions in genome-wide case-control studies," *The American Journal of Human Genetics*, vol. 87, no. 3, pp. 325–340, 2010.

[142] H. Wang *et al.*, "Genome-wide epistasis analysis for alzheimer's disease and implications for genetic risk prediction," *Alzheimer's Research & Therapy*, vol. 13, no. 1, p. 55, 2021.

[143] X. Wang, E. P. Xing, and D. J. Schaid, "Kernel methods for large-scale genomic data analysis," *Briefings in Bioinformatics*, vol. 16, no. 2, pp. 183–192, 07 2014. [Online]. Available: https://doi.org/10.1093/bib/bbu024

[144] Z. Wang, J. H. Sul, S. Snir, J. A. Lozano, and E. Eskin, "Gene-gene interactions detection using a two-stage model," *Journal of Computational Biology*, vol. 22, no. 6, pp. 563–576, 2015, pMID: 25871811. [Online]. Available: https://doi.org/10.1089/cmb.2014.0163

[145] Z. Wang, Y. Wang, K.-L. Tan, L. Wong, and D. Agrawal, "eCEO: an efficient Cloud Epistasis cOmputing model in genome-wide association study," *Bioinformatics*, vol. 27, no. 8, pp. 1045–1051, 03 2011. [Online]. Available: https://doi.org/10.1093/bioinformatics/btr091

[146] W. H. Wei, G. Hemani, and C. S. Haley, "Detecting epistasis in human complex traits," *Nature Reviews Genetics*, vol. 15, no. 11, pp. 722–733, 2014.

[147] L. Wienbrandt, J. C. Kässens, and D. Ellinghaus, "Snpint-gpu: Tool for epistasis testing with multiple methods and gpu acceleration," in *Epistasis: Methods and Protocols*, K.-C. Wong, Ed., vol. 2212. New York, NY: Springer US, 2021, pp. 17–35. [Online]. Available: https://doi.org/10.1007/978-1-0716-0947-7_2

[148] L. Wienbrandt, J. C. Kässens, M. Hübenthal, and D. Ellinghaus, "1000x faster than plink: Combined fpga and gpu accelerators for logistic regression-based detection of epistasis," *Journal of Computational Science*, vol. 30, pp. 183–193, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877750318310184

[149] J. Wu, B. Devlin, S. Ringquist, M. Trucco, and K. Roeder, "Screen and clean: a tool for identifying interactions in genome-wide association studies," *Genetic Epidemiology*, vol. 34, no. 3, pp. 275–285, 2010. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/gepi.20459

[150] M. C. Wu, A. Maity, S. Lee, E. M. Simmons, Q. E. Harmon, X. Lin, S. M. Engel, J. J. Molldrem, and P. M. Armistead, "Kernel machine snp-set testing under multiple candidate kernels," *Genetic Epidemiology*, vol. 37, no. 3, pp. 267–275, 2013. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/gepi.21715

[151] T. Yanes, M.-A. Young, B. Meiser, and P. A. James, "Clinical applications of polygenic breast cancer risk: a critical review and perspectives of an emerging field," *Breast Cancer Research*, vol. 22, no. 1, p. 21, 2020.

[152] H. Yang, S. Li, H. Cao, C. Zhang, and Y. Cui, "Predicting disease trait with genomic data: a composite kernel approach," *Briefings in Bioinformatics*, vol. 18, no. 4, pp. 591–601, 06 2016. [Online]. Available: https://doi.org/10.1093/bib/bbw043

[153] P. Yang, J. Ho, Y. H. Yang, and B. B. Zhou, "Gene-gene interaction filtering with ensemble of filters," *BMC Bioinformatics*, vol. 12, no. 1, p. S10, 2011.

[154] P. Yang, B. B. Zhou, J. Y.-H. Yang, and A. Y. Zomaya, "Stability of feature selection algorithms and ensemble feature selection methods in bioinformatics," in *Biological Knowledge Discovery Handbook*. John Wiley & Sons, Ltd, 2013, pp. 333–352. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118617151.ch14

[155] L. S. Yung, C. Yang, X. Wan, and W. Yu, "GBOOST: a GPU-based tool for detecting gene–gene interactions in genome–wide case control studies,"

Bioinformatics, vol. 27, no. 9, pp. 1309–1310, 03 2011. [Online]. Available: https://doi.org/10.1093/bioinformatics/btr114

[156] N. Zaitlen and P. Kraft, "Heritability in the genome-wide association era," *Human Genetics*, vol. 131, no. 10, pp. 1655–1664, 2012.

[157] H. Zhang, P. Meltzer, and S. Davis, "Rcircos: an r package for circos 2d track plots," *BMC Bioinformatics*, vol. 14, no. 1, p. 244, 2013.

[158] H. Zhang, G. Yu, W. Ren, M. Guo, and J. Wang, "Epintmc: Detecting epistatic interactions using multiple clusterings," in *Bioinformatics Research and Applications*, Z. Cai, I. Mandoiu, G. Narasimhan, P. Skums, and X. Guo, Eds. Cham: Springer International Publishing, 2020, pp. 56–67.

[159] X. Zhang, S. Huang, F. Zou, and W. Wang, "TEAM: efficient two-locus epistasis tests in human genome-wide association study," *Bioinformatics*, vol. 26, no. 12, pp. i217–i227, 06 2010. [Online]. Available: https://doi.org/10.1093/bioinformatics/btq186

[160] X. Zhang, F. Pan, Y. Xie, F. Zou, and W. Wang, "Coe: A general approach for efficient genome-wide two-locus epistasis test in disease association study," *Journal of Computational Biology*, vol. 17, no. 3, pp. 401–415, 2010, pMID: 20377453. [Online]. Available: https://doi.org/10.1089/cmb.2009.0155

[161] A. Ziegler and I. R. König, *A Statistical Approach to Genetic Epidemiology: Concepts and Applications*, 2nd ed. Weinheim: WILEY-VCH Verlag GmbH & Co. KGaA, 2010.

[162] A. Ziegler, I. R. König, and J. R. Thompson, "Biostatistical aspects of genome-wide association studies," *Biometrical Journal*, vol. 50, no. 1, pp. 8–28, 2008. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/bimj.200710398